



OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

FAKULTÄT FÜR
MATHEMATIK




Deutsches Zentrum
für Luft- und Raumfahrt

Institut für Softwaretechnologie

Combinatorial Problems in Programming Quantum Annealers

Dissertation zur Erlangung des akademischen Grades



**doctor rerum naturalium
(Dr. rer. nat.)**

von M.SC. ELISABETH LOBE ,
geboren am 17.01.1990 in Meißen,

genehmigt durch die Fakultät für Mathematik
der Otto-von-Guericke-Universität Magdeburg,

begutachtet von PROF. DR. VOLKER KAIBEL und
PROF. DR. FRAUKE LIERS,

eingereicht am 25. Januar 2022,
verteidigt am 13. Mai 2022,

betreut am Institut für Softwaretechnologie des
Deutschen Zentrums für Luft- und Raumfahrt
von DR. TOBIAS STOLLENWERK  und
DR.-ING. ACHIM BASERMANN 

Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht.

Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert oder verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Declaration of Honor

I hereby declare that I produced this thesis without prohibited assistance and that all sources of information that were used in producing this thesis, including my own publications, have been clearly marked and referenced.

In particular, I have not wilfully:

- Fabricated data or ignored or removed undesired results.
- Misused statistical methods with the aim of drawing other conclusions than those warranted by the available data.
- Plagiarised data or publications or presented them in a distorted way.

I know that violations of copyright may lead to injunction and damage claims from the author or prosecution by the law enforcement authorities.

This work has not previously been submitted as a doctoral thesis in the same or a similar form in Germany or in any other country. It has not previously been published as a whole.

Magdeburg, 25. Januar 2022

Elisabeth Lobe

Zusammenfassung

Bevor auf einer Quanten-Annealing-Maschine, wie der der Firma D-Wave Systems Inc., Berechnungen durchgeführt werden können, sind zwei grundlegende Schritte notwendig, um das Originalproblem in ein Format zu übertragen, das von solchen Maschinen gelöst werden kann: Als Erstes muss ein mit dem Problem assoziierter Graph in den speziellen Hardwaregraphen eingebettet werden und als Zweites müssen die Parameter des eingebetteten Problems entsprechend weiterer Hardwarerestriktionen gewählt werden, sodass die Lösungen des eingebetteten Problems beweisbar äquivalent zu den ursprünglichen Lösungen sind. Diese Doktorarbeit adressiert graphentheoretische Fragestellungen und kombinatorische Optimierungsprobleme, die bei der genaueren Betrachtung beider Schritte auftreten.

Im ersten Teil dieser Arbeit analysieren wir die Komplexität des Einbettungsproblems im Quanten-Annealing-Kontext, das heißt für Chimera- und Pegasus-Hardwaregraphen mit zum Teil nicht nutzbaren, „defekten“ Qubits. Wir beweisen die Schwere des Hamiltonkreisproblems, einem Spezialfall des Einbettungsproblems, in solchen Graphen durch die Konstruktion defekter Chimera-Graphen aus speziellen Graphen, für welche die Schwere des Problems bereits bekannt ist. Da der Chimera- ein Subgraph des Pegasus-Graphen ist, können wir das Resultat auf letzteren übertragen.

Ein weiterer Spezialfall ist die Einbettung eines vollständigen Graphen, welcher ein universelles Template für die Einbettung von beliebigen Graphen mit einer kleineren oder gleich großen Zahl an Knoten darstellt. Durch die Formulierung als Matchingproblem mit zusätzlichen linearen Nebenbedingungen können wir zeigen, dass das Problem eingeschränkt auf die sich natürlich ergebende Einbettungsstruktur „fixed-parameter tractable“ ist, wenn wir die Zahl der defekten Qubits im Chimera-Graphen als Parameter betrachten. Wir vergleichen unser Verfahren mit vorherigen, heuristischen Ansätzen auf verschiedenen, zufällig generierten defekten Hardwaregraphen. Dabei können wir einen Vorteil unserer Methode gegenüber den anderen hinsichtlich der gefundenen Graphengrößen in der Praxis zeigen. Zusätzlich geben wir ein heuristisches Modell mit weniger Nebenbedingungen an, welches noch bessere Resultate liefert.

Der zweite Teil beschäftigt sich mit der Wahl der geeigneten Parameter, für welche wir hinreichende Bedingungen formulieren können. Durch die Betrachtung eines einzelnen Originalknotens und verschiedener, von der Hardware abgeleiteter Zielsetzungen können wir spezielle lineare Optimierungsprobleme extrahieren. Die Analyse eines entsprechenden Polyeders der zulässigen Lösungen zeigt, dass optimale Lösungen zu diesen Problemen in vielen Fällen in Linearzeit gefunden werden können. Für die verbleibenden Fälle konstruieren wir einen Algorithmus, der die Parameter in höchstens kubischer Laufzeit angibt. Aufgrund der Problemstruktur gelten diese Resultate sogar, wenn wir uns auf Ganzzahligkeit einschränken.

Abstract

Before being able to perform calculations on a quantum annealing device such as D-Wave's, two essential steps are required to transfer the original problem into a format which can be solved by these machines: First, a graph associated with the problem needs to be embedded into the specific hardware graph and, secondly, the parameters of the embedded problem need to be chosen, in accordance with further hardware restrictions, such that the solutions to the resulting problem are provably equivalent to those of the original problem. This thesis addresses graph theoretical questions and combinatorial optimization problems appearing in the closer examination of both steps.

In the first part of this work, we analyze the complexity of the embedding problem in the quantum annealing context, this means when restricting to Chimera or Pegasus hardware graphs containing unavailable, 'broken' qubits. We prove the hardness of the Hamiltonian cycle problem, a special case of the embedding problem, in such graphs by constructing broken Chimera graphs from certain graphs for which it is known that finding a Hamiltonian cycle is hard. As the Chimera graph is a subgraph of the Pegasus graph, we can easily extend the result to the latter.

A further special case is the embedding of a complete graph, forming a universal template for the embedding of all arbitrary graphs with a smaller or equal number of vertices. By formulating this problem as a matching problem with additional linear constraints, we can prove that the problem restricted to the naturally arising embedding structure is fixed-parameter tractable in the number of broken vertices in the Chimera graph. By testing our model against previous, heuristic approaches on various random broken hardware graphs, we can further show that our method performs superior in practice. Additionally, we provide a heuristic model with less constraints, showing an even better performance.

The second part is concerned with the problem of setting feasible parameters for the machine, for which we can formulate sufficient requirements. Considering a single original vertex and different objectives derived from the hardware restrictions, we extract certain linear optimization problems. By analyzing a corresponding polyhedron of feasible solutions, we can show that optimal solutions to these problems can be found in linear time for a lot of cases. For the remaining cases, we construct an algorithm providing the parameters in at most cubic time. Due to the problem structure, these results even hold if we restrict ourselves to integer problems.

Acknowledgments

I gratefully acknowledge the Jülich Supercomputing Centre (JSC) for supporting this work by providing the access to the D-Wave 2000Q through the Jülich UNified Infrastructure for Quantum computing (JUNIQ).

Furthermore, I am very thankful for the working conditions and the support that I found at the German Aerospace Center (DLR), where I prepared this thesis at the Institute for Software Technology.

Contents

1	Introduction	1
1.1	The Black Box	1
1.2	Programming a Quantum Annealer	3
1.3	Outline	5
2	Restricted Ising Problem over Chimera Graph	7
2.1	General Notation	7
2.2	Ising Problem	8
2.3	Chimera Graph	11
3	Minor Embedding	13
3.1	Minors and Embeddings	13
3.1.1	Basic Definitions	13
3.1.2	In Quantum Annealing	15
3.1.3	Related Work	16
3.2	NP-Completeness Proof	18
3.2.1	Reduction of the Hamiltonian Cycle Problem	18
3.2.2	Basics of Grid Graphs	20
3.2.3	Broken Chimera Graph Construction	22
3.2.4	Hamiltonicity	27
3.2.5	Transfer to Pegasus	40
3.3	Largest Complete Graph	43
3.3.1	Matching Problem Approach	43
3.3.2	Extension of the Chimera Graph Description	45
3.3.3	ILP Formulation	47
3.3.4	Analysis	53
3.3.5	Heuristic ILP	57
3.3.6	Experimental Setup	58
3.3.7	Results	59
4	Weight Distribution	63
4.1	Problem Extraction	63
4.1.1	Embedded Ising Model	63
4.1.2	Synchronization	66
4.1.3	Related Work	68
4.1.4	Single Vertex Evaluation	69
4.2	Optimization Problem Formulation	71
4.2.1	Instance Definition	71
4.2.2	Simplifications	75
4.2.3	LP Formulation	82
4.2.4	Case Distinction	85

4.3	Special Cases	87
4.3.1	Trivial Weighting	87
4.3.2	Zero Weight	88
4.4	Simplified Description of the Θ -Polyhedron	92
4.4.1	Connected Vertex Sets	92
4.4.2	Arcs in Trees	98
4.4.3	Eliminating the Gap Requirement	100
4.4.4	Unique Root	102
4.4.5	Contraction of Multiple Roots	104
4.5	Strength-Only Problem on Trees	106
4.5.1	Objective Value	106
4.5.2	Straightforward Weights	109
4.5.3	Second-level Weight Optimization	112
4.5.4	De-contraction	119
4.6	Full Problem on Trees	122
4.6.1	Uniqueness	123
4.6.2	Non-negative Weights	125
4.6.3	Weight Calculation Algorithm	130
4.6.4	Adjusted Algorithm Including De-contraction	142
4.7	Integer Programming	147
4.7.1	The Problems	147
4.7.2	Straightforward Insights	148
4.7.3	Rounded Up	149
5	Conclusions	155
	Lists	III
	Problems	III
	Figures	III
	Tables	V
	Algorithms	VI
	Bibliography	VII

1 Introduction

Quantum computing devices have gained growing attention in recent years. By the exploitation of quantum mechanical effects, this new technology promises to defeat classical computers in solving difficult problems, at least prospectively with the ongoing development of these machines: While a classical bit can only be in one of two possible states, ‘0’ or ‘1’, a quantum bit, in short *qubit*, can be in two states at the same time. This property, called *superposition*, enables the *quantum parallelism*, which allows to calculate on both values at the same time.

Even more such powerful quantum mechanical properties exist, which draw a distinction between the quantum and the classical world. By transferring their advantages to new models of computation, new machines shall be built that execute computations based upon these properties more efficiently than classical devices. A decisive step has been done by the Canadian company D-Wave Systems Inc. with the development of the first commercially available quantum annealer.

1.1 The Black Box

The quantum device of D-Wave implements a specific physical model, the *Ising model*, named after the German physicist Ernst Ising, who investigated the ferromagnetism in solids together with his doctoral advisor Wilhelm Lenz [27]. The quantum processor of such a machine consists of overlapping superconducting loops, in which the direction of a current flow causes a magnetic spin. This spin points up or down, thus is in state ‘+1’ or ‘-1’, or is in a quantum mechanical superposition of both, thus in ‘+1’ and ‘-1’ simultaneously. Due to this feature, the superconducting loops can represent qubits in a quantum system [29].

Where two loops are coupled by a joint, they influence each other in one of two possible ways: If the coupling is ferromagnetic, the spins of both loops tend to point in the same direction, if it is antiferromagnetic, they point to different directions. By applying an external magnetic field, the quantum annealer can adjust the magnetism. The pairwise interactions between all qubits can therefore be described by a quadratic function with quadratic terms for all interacting loops weighted depending on the magnetism. The minimal solution of this function corresponds to the state of the quantum system with the lowest possible energy, the *ground state*. In other words, we can construct a quantum system that represents the objective function of a quadratic binary optimization problem and, in order to solve this problem, we need to prepare the ground state.

D-Wave applies the following strategy: The machine starts in a configuration of the quantum system whose ground state can easily be set up and is formed by all qubits being in a superposition. Afterwards, the magnetic fields are modified to yield the desired quantum system, meanwhile the superpositions may break up. If the manipulation is done ‘sufficiently slow’, the resulting system is still in the ground state and, by measuring the system, we obtain the (classical) optimal spin states according to the magnetism and with them an optimal solution to the encoded optimization problem.

‘Sufficiently slow’ means here by an *adiabatic evolution process* in accordance with the *adiabatic theorem*, a fundamental result in quantum mechanics, stated in its original form in [8]. Therefore, the underlying concept is also called *adiabatic quantum computation* [19]. This needs however to be distinguished from ‘conventional’ quantum computation because it does not yield universal programmability in a straightforward way, which means in terms of implementing quantum circuits with quantum logical gates. Corresponding machines are rather built for executing only a single ‘algorithm’. Representing a natural process, the devices are sometimes also referred to as analog quantum computers [17]. Nevertheless, in [3] the authors could show that adiabatic quantum computation can efficiently simulate quantum circuits by encoding their output as the final ground state. For detailed background information about the computational concept, we refer to the comprehensive review in [4].

The adiabatic theorem is only applicable under ideal conditions. The realization of this theoretical concept however turns out to be difficult due to several physical restrictions. One of them is, for instance, the shielding against noise from the environment, which can never be achieved entirely. Thus, the D-Wave machines in reality rather serve as a sampler of the low-energy distribution of the desired quantum system: By repeating the adiabatic evolution process several times with the same configuration, various solutions are obtained in a single run [31].

Although some empirical studies, such as [30], show that the computational output is in general close to the optimal objective value, the probability of finding the optimal solution at least once in such a run varies greatly and cannot be determined in advance. Therefore, these ‘imperfect’ devices can only be considered as heuristic solvers. For the distinction from the computational concept, the term *quantum annealing*, with reference to the classical heuristic *simulated annealing*, has been established nowadays to describe the process of heuristically solving optimization problems through adiabatic evolution [41].

Several factors can suppress what is known as the *success probability* of the quantum annealers even further. Some of them can be influenced by user-defined system parameters, such as the annealing time. Those are listed and explained in detail in the documentation of D-Wave [14]. However, it is in general hard to predict which choices for these parameters yield the best performance for a specific problem because they strongly depend on it.

In any case, the D-Wave machines can only process a specific problem, a restricted version of what is called, in analogy to the physical concept, the *Ising problem*. We explain it in detail in Chapter 2. Several studies, such as [45, 48, 49, 51], have revealed that interesting applications usually do not match this kind of problem structure, which is why several transformation steps are required in advance to transfer the actual problem to the specific format, adding another level of complexity. The process of determining the final input parameters of the optimization problem instance is what we call the *programming* of the quantum annealer.

This is the point where this thesis builds upon and extends the current developments. While the step from an arbitrary combinatorial optimization problem to a general Ising problem can easily be done using standard techniques of mathematical optimization, see e.g. [38], the next step to the restricted Ising problem leaves some open questions and unsolved problems, although the limitations are quite well examined and understood [31]. These gaps need to be closed before the user can perform useful calculations on the machine and understand its advantages and disadvantages over classical approaches. This means, in this work, we focus on the specific programming restrictions, but apart from that, we consider the annealing machines as a black box without questioning their ability to actually solve the programmed problems. In Figure 1.1, we illustrate the abstraction steps that usually precede calculations on a D-Wave machine and highlight the circle this work aims to close.

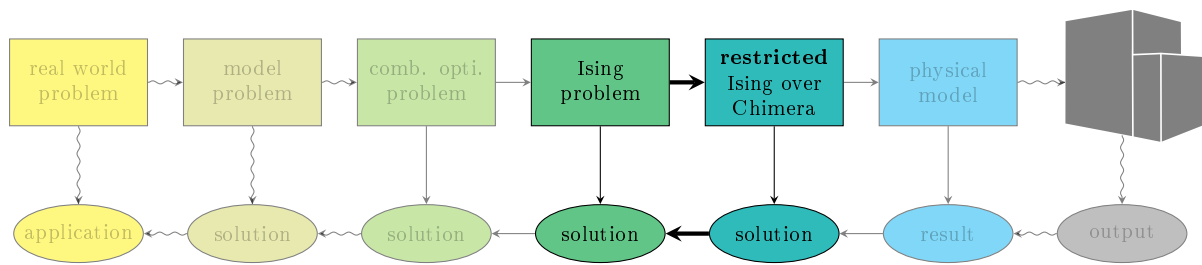


Figure 1.1: Layers of abstraction in transferring a problem to D-Wave’s quantum annealer, depicted in upper right corner

1.2 Programming a Quantum Annealer

The main reference for programming quantum annealers is still the work of V. Choi published in 2008 and 2011 [12, 13]. There the author has presented the two main steps, the *minor embedding* and the *parameter setting*, and how they could be approached. Although the machines have developed over time, these steps have consisted due to the general underlying structure of these machines.

To run a problem on the D-Wave machine, the user simply sends the coefficients of the objective function of the Ising problem over the programming interface. However, due to the physical layout of the superconducting loops, we do not have an all-to-all connectivity between the qubits and thus only a limited structure for non-zero coefficients. The available quadratic monomials, corresponding to two interacting qubits each, form specific graphs, for which D-Wave introduced the terms *Chimera* and *Pegasus* graph [6]. The latter describes the most recent architecture, which was released in 2020 and is derived from the Chimera structure.

An Ising problem with a different coefficient structure is however associated with a different graph. This means, in order to solve such an arbitrary problem with the quantum annealer, we need to simulate the desired connectivity of the problem graph with the limited one of the hardware graph. This is done by what is known as a *minor embedding*: A vertex of the problem graph is mapped to several vertices of the hardware graph. We give a precise definition of such an embedding in Section 3.1.1.

The Chimera graph was designed to yield a simple straightforward embedding of the complete graph [13], which enables to embed all graphs with a less or equal number of vertices. Due to the regular underlying structure of both, the Chimera graph and the embedding instructions, this layout can easily be extended for growing hardware graphs. While the number of vertices in the complete graph grows linearly, the number of required vertices in the Chimera however grows quadratically. For example, in the currently operating DW_2000Q solver of D-Wave, we can therefore embed only up to 65 completely connected vertices, although it contains 2048 qubits [32]. This overhead is a general problem of the annealers, requiring the actual problem to be significantly smaller than what the hardware parameters promise at first sight.

Another limiting factor is that the regular built-in hardware structure does not transfer one-to-one to the available hardware graph. Some qubits, or in rare cases the couplings between them, are inaccessible through the programming interface. They are taken offline because they do not operate in the expected way [14]. With every calibration of the machine, the location of such

‘broken’ qubits changes unpredictably. However, recalibration is only necessary in the order of months or even years, which means the hardware graphs remain over this period.

As a result of these broken vertices, standard embedding schemes do not apply anymore. Those are nevertheless only available for a few graphs closely related to the Chimera structure [35]. This means we always need to solve the embedding problem before being able to run experiments on the quantum annealer. Even with the ongoing development of the machines, we do not expect the problem to vanish. The development of an all-to-all connectivity is unlikely due to several physical limitations and, with an increasing total number of qubits, the probability of a broken qubit will probably decrease but not be equal to 0. The embedding problem will thus remain relevant for calculations with the annealing machine in the long-term.

After the embedding has been found, the parameters of the Ising problem which shall be solved originally need to be transferred to the new coefficient structure. The resulting *embedded Ising problem* should indeed represent the original Ising problem such that the corresponding solutions can (in theory) be retrieved from the output of the quantum annealer. This non-trivial problem of finding suitable parameters for the provable equivalence of the problems is called by V. Choi the *parameter setting* [12].

Unfortunately, we need to take further restrictions on the parameters into account. First of all, they can only be chosen within a certain interval, where the concrete boundary values might vary between the different architectures. At first sight, this does not seem to be problematic: We can simply scale the whole objective function of the Ising problem by multiplying a constant factor. This however decreases the absolute difference between the coefficients.

The parameter precision is the most critical restriction of D-Wave’s annealing machines. Due to the transmission over the analog control circuits, the problem defining parameters experience different perturbations [31]. This means that the actually solved problem differs slightly from the one specified by the user. Thus, problems which shall be solved with these machines need to be chosen carefully to yield some kind of ‘robustness’ in the parameter precision.

Although the programming interface allows to insert arbitrary float values within given ranges, the machine can actually realize only a limited discrete coefficient range [30]. In [49], a precision of about $\frac{1}{30}$ was estimated for the specific annealer used in the experiments. For problems with a higher precision, the success probability is drastically reduced because the annealing machine is not capable of resolving the parameters. In more recently released machines, the precision is probably improved. However, the specific values are not precisely known and can only be estimated through further experiments.

For a user, the programming of such annealing machines is only worth the effort if the machine can find the optimal solution to the provided problem in a certain number of runs, that is, if an acceptable success probability can be achieved. This in turn requires the models to possess certain properties concerning their defining parameters. As the boundaries of the necessary parameters cannot be specified exactly, we can rather formulate some objectives aiming to improve the success probability as much as possible.

A first step when encoding a problem as an Ising problem is therefore to keep the largest appearing coefficients as small as possible (without scaling). In presence of a very large coefficient, two others might appear too close to each other for the machine. This becomes particularly important when Ising models on other graphs than the hardware graphs shall be solved: Such large values usually appear with the ‘strong coupling’ of the vertices in the embedding of a single original

vertex, where the corresponding quadratic terms obtain large absolute coefficients to enforce that the qubits behave collectively during the annealing process.

Furthermore, the former considerations support the assumption that it is preferable to have as few different values for the coefficients in the Ising model as possible, while those should have the largest possible pairwise absolute difference to achieve an acceptable success probability. Rather than dealing with rational coefficients, a possible approach would be to fix the distance to 1 by only allowing for integer parameters in a certain range. Again the concrete bounds are not precisely known. Therefore, the largest absolute integer coefficient should be minimized.

1.3 Outline

In this thesis, we address both programming steps, the minor embedding and the parameter setting, as they still form a bottleneck preventing successful computations on the machines, even before the problems are sent to these machines.

We start with introducing the specific optimization problem the quantum annealers can process in Chapter 2. Afterwards, we investigate the embedding problem in detail in Chapter 3 and prove hardness results for the corresponding embedding problems which are relevant in the quantum annealing context. Furthermore, we provide optimization problem formulations to approach the special case of the complete graph embedding in the broken Chimera graph.

In Chapter 4, the parameter setting problem is approached by the derivation of different optimization problems from the sufficient requirements on the coefficients of the embedded Ising model. These problems are analyzed in detail and, after the description of an efficient algorithm to calculate the coefficients, we transfer the results into the integer programming framework to overcome the precision issue of the machines. Finally, we conclude our work in Chapter 5.

2 Restricted Ising Problem over Chimera Graph

D-Wave's quantum annealing machines can only solve a very restricted problem, which we introduce in detail in this section. We start with the general notation used throughout this thesis in Section 2.1. After presenting the general mathematical Ising model in Section 2.2 with the corresponding parameter requirements, we present the specific graph derived from the hardware structure in Section 2.3, to which the solvable Ising models are restricted.

2.1 General Notation

First, we introduce some general notations used throughout this work. For some $n, m \in \mathbb{N}$, let $[m; n] := \{m, m + 1, \dots, n\}$ be the enumeration from m to n , where we have $[m; n] = \emptyset$ for $n < m$. For shortness, we use $[n] := [1, n]$ for enumerating from 1, where we say $[0] = \emptyset$. If a set S is the disjoint union of two sets S_1 and S_2 , that means $S_1 \cup S_2 = S$ and $S_1 \cap S_2 = \emptyset$, we use $S = S_1 \cup S_2$. With 2^X we denote the set of all subsets of a set X .

For the basic graph definitions, we generally follow the standard literature in graph theory and optimization, see e.g. [18] or [34], and briefly recapture the main notations here: With $G = (V, E)$ we always refer to a simple undirected finite graph with the finite set of vertices V and the set of edges $E \subseteq \{\{v, w\} : v, w \in V\}$. Given a graph G , $V(G)$ and $E(G)$ provide the vertex and the edge set, respectively, if those are not named specifically. While a subgraph of G is formed by arbitrary subsets of edges and vertices of G , $G[S]$ for some vertex set $S \subset V(G)$ refers to the vertex-induced subgraph of graph G , where we have $V(G[S]) = S$ and $E(G[S]) = \{\{v, w\} \in E(G) : v, w \in S\}$. With K_n and $K_{n,n}$, we denote the complete graph with n vertices and the complete bipartite graph with n vertices in each partition, respectively.

For shortness, we abbreviate an edge $\{v, w\}$ with the commutative product vw . Where applicable, we also identify the tuple $(x, y) \in X \times Y$ for two sets X and Y with xy , although x and y do not commute in this case. In general, it is clear from the context whether reverting the product ordering is feasible. As a rule of thumb, we use xy for integer coordinates in the two-dimensional space, while vw refers to an edge of a graph.

We denote the neighbors of a vertex v in the graph G with

$$N_G(v) := \{w \in V(G) : vw \in E(G)\},$$

where we drop the subscript G when it is clear from the context that we have $v \in V(G)$. The incident edges are

$$\begin{aligned} \delta_G(S) &:= \{vw \in E(G) : v \in S, w \in V(G) \setminus S\}, \\ \delta_G(S, T) &:= \{vw \in E(G) : v \in S, w \in T\} = \delta_G(S) \cap \delta_G(T), \end{aligned}$$

where we again drop the subscript G when it is clear that we have $S, T \subseteq V(G)$. We use $\delta(v)$ to abbreviate $\delta(\{v\})$.

For indexed parameters or variables $x \in X^I$ with the index set I and the value set X , we use $x_J = (x_i)_{i \in J}$ for a subset $J \subseteq I$ of the indices to refer to a subset of these parameters or variables, respectively, the corresponding vector. In turn, we ‘apply’ J by

$$x(J) = \sum_{i \in J} x_i.$$

We denote the vector containing only 1’s or 0’s by $\mathbb{1}$ and $\mathbb{0}$, respectively. For both, we add the subscript for the corresponding index set wherever necessary.

We further use the Big-O notation $\mathcal{O}(f)$ for the set of functions that are asymptotically bounded from above by a constant times the function f .

Finally, we give a short style guide: New and important terms are highlighted in *italic*. Newly defined mathematical symbols are marked in **bold** on their first appearance and indicated with ‘:=’. Additionally, if some of the introduced symbols are considered to be given and fixed over a longer textual part, we mention and highlight them accordingly in the beginning of that section. Apart from the global enumeration of referenced equations, we use a local enumeration with small roman letters, for example (i) - (iii), inside of proofs, which is not referenced outside.

2.2 Ising Problem

In the quantum annealing processor, the magnetism of the loops and their couplings can be adjusted with user-defined input parameters. This means we can encode different quadratic functions. The term ‘Ising model’ also refers to these objective functions because they are closely related to the formulation of the physical model [12]. We use throughout this work:

Definition 2.1. An *Ising model* over graph G with *weights* $\mathbf{W} \in \mathbb{R}^{V(G)}$ and *strengths* $\mathbf{S} \in \mathbb{R}^{E(G)}$ is a function $\mathbf{I}_{\mathbf{W}, \mathbf{S}} : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ with

$$I_{\mathbf{W}, \mathbf{S}}(s) := \sum_{v \in V(G)} W_v s_v + \sum_{vw \in E(G)} S_{vw} s_v s_w.$$

We call G the *interaction graph* of the Ising model.

Usually, we keep the interaction graph fixed. To be able to differ between two Ising models for the same graph, we use the symbol $I_{\mathbf{W}, \mathbf{S}}$ with the corresponding weights and strengths in the subscript. In case those are clear from the context, we drop the subscript.

Using this definition, we can formulate a general version of the optimization problem the quantum annealing machine can process:

ISING PROBLEM. Given a graph G , $\mathbf{W} \in \mathbb{R}^{V(G)}$ and $\mathbf{S} \in \mathbb{R}^{E(G)}$, find s that solves

$$\min_{s \in \{-1, 1\}^{V(G)}} I_{\mathbf{W}, \mathbf{S}}(s).$$

D-Wave's quantum annealer can indeed only implement float values with $W \in [-m, m]^{V(G)}$ and $S \in [-n, n]^{E(G)}$ for specific $m, n \in \mathbb{N}$. For instance, for the current Chimera architecture, we have $m = 2$ and $n = 1$. However, due to possible scaling, this is not a hard restriction. A value which provides more insight in the coefficient distribution is the maximal absolute coefficient

$$\mathbf{C}_{\max} := \max \{ \|W\|_{\infty}, \|S\|_{\infty} \} = \max \left\{ \max_{v \in V(G)} |W_v|, \max_{vw \in E(G)} |S_{vw}| \right\},$$

in particular when compared with its counterpart, the minimal absolute coefficient being unequal to zero, or the minimal difference between two absolute coefficients

$$\min \{ |x - y| : x \neq y \in \{0\} \cup \{W_v : v \in V(G)\} \cup \{S_{vw} : vw \in E(G)\} \}.$$

If we further restrict the weights and strengths to \mathbb{Z} according to the differentiation considerations, which means on the integer range $[-m; m] = \{-m, -m + 1, \dots, m\}$, respectively, $[-n; n]$, the latter becomes 1 after scaling. Thus, the maximal absolute coefficient C_{\max} is a decisive value to estimate whether the problem is suitable to be solved with the annealer. According to [49], we need at least $C_{\max} \leq 30$ to achieve an acceptable success probability.

The decision problem corresponding to the ISING PROBLEM is known to be NP-complete [5]. This means a variety of problems can be mapped to it in polynomial time [38]. In particular, it is closely related to the QUADRATIC UNCONSTRAINED BINARY OPTIMIZATION PROBLEM (QUBO), more commonly known and well studied in combinatorial optimization. Such problems are defined analogously to the Ising problems apart from working on binary variables $x \in \{0, 1\}^{V(G)}$. The objective function is then also called a quadratic pseudo-Boolean function. The corresponding broader class of pseudo-Boolean optimization was extensively studied by E. Boros and P. L. Hammer, for instance, in [10].

Both problems, QUBO and Ising, can easily be transferred into each other with the affine transformation $x_v = \frac{1}{2}(s_v + 1)$ for $v \in V(G)$. This changes the weights and strengths of the objective function, nevertheless, the graph defined by the non-zero strengths remains the same. A possibly appearing constant can be extracted from the optimization problem, but surely needs to be remembered in case the original objective value is of further interest.

Apart from some direct mappings, for instance, as listed in [38], the most common way to obtain an Ising model starts with an optimization problem, which is usually transformed to a QUBO in the intermediate step. The reduction steps are well known and mainly consist of

- the encoding of the integer variables as binary variables,
- the reduction of the degree by the introduction of new variables being enforced to represent a product of variables,
- the transformation of inequalities to equalities by the use of slack variables and
- the reduction of equality constraints by adding penalty terms to the objective function.

See, for example, [33] for more details.

Usually, there are several possibilities how to combine the above transformation steps, resulting in different Ising models equivalently representing the same original problem. As all of these steps can also influence C_{\max} decisively, they however might need to be chosen with great care to obtain an Ising model that fits the requirements of the annealing hardware.

Due to the close relation of QUBO and Ising problems, we could apply preprocessing steps as introduced in [9] to the corresponding QUBO to simplify the considered model in advance. However, there are similar preprocessing methods for directly manipulating the Ising model. One of them is applicable if the weight of a vertex exceeds the influence of the strengths of the incident edges. We recall the well known result here for completeness because it implies the exclusion of a certain case in the investigations in Chapter 4.

Lemma 2.2. *For an Ising model $I_{W,S} : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ over a graph G with $W \in \mathbb{R}^{V(G)}$ and $S \in \mathbb{R}^{E(G)}$, if we have*

$$|W_v| > \sum_{n \in N(v)} |S_{vn}|$$

for some vertex $v \in V(G)$, every optimal solution

$$s^* \in \arg \min_{s \in \{-1, 1\}^{V(G)}} I_{W,S}(s)$$

fulfils $s_v^* = -\text{sign}(W_v)$.

Proof. We extract the part of $I_{W,S}$ containing s_v^* with

$$I_{W,S}(s^*) = \sum_{w \in V(G) \setminus \{v\}} W_w s_w^* + \sum_{wu \in E(G) \setminus \delta(v)} S_{wu} s_w^* s_u^* + \underbrace{W_v s_v^* + \sum_{n \in N(v)} S_{vn} s_v^* s_n^*}_{=: I^v(s_v^*)}$$

where we keep the other s -variables apart from s_v^* fixed. With the condition for vertex v , we have

$$|W_v| > \sum_{n \in N(v)} t_n S_{vn} \quad \forall t \in \{-1, 1\}^{N(v)}$$

and therefore can observe that

$$\begin{aligned} I^v(\text{sign}(W_v)) &= |W_v| + \sum_{n \in N(v)} S_{vn} (\text{sign}(W_v) s_n^*) \\ &> 0 \\ &> -|W_v| + \sum_{n \in N(v)} S_{vn} (-\text{sign}(W_v) s_n^*) \\ &= I^v(-\text{sign}(W_v)). \end{aligned}$$

This shows that the contribution of $s_v^* = \text{sign}(W_v)$ is always larger than the negated choice independently of the assignment of the other s -variables. \square

Remark: It is also easy to see that, if the equality holds in the above condition for v , the optimal solution does not necessarily hold the value $-\text{sign}(W_v)$ for s_v^* . Still, this only happens if the last inequality in the proof collapses to an equality. Therefore, both choices yield the same optimal value and we can nevertheless choose to set $s_v^* = -\text{sign}(W_v)$ in advance.

Based on this result, we can formulate a simpler Ising model: By fixing s_v to the predetermined value $-\text{sign}(W_v)$, with v of the above lemma, but keeping the other variables arbitrary, the former quadratic terms corresponding to the incident edges become linear and therefore integrate into

the weights of the neighbors of v . We get the new Ising model $I_{\bar{W}, \bar{S}} : \{-1, 1\}^{V(G) \setminus \{v\}} \rightarrow \mathbb{R}$ with $W \in \mathbb{R}^{V(G) \setminus \{v\}}$, $S \in \mathbb{R}^{E(G) \setminus \delta(v)}$ and

$$\begin{aligned} \bar{W}_w &= W_w & \forall w \in V(G) \setminus (\{v\} \cup N(v)), \\ \bar{W}_w &= W_w - \text{sign}(W_v) S_{vw} & \forall w \in N(v), \\ \bar{S}_e &= S_e & \forall e \in E(G) \setminus \delta(v). \end{aligned}$$

Note that, in this new model, the constant $|W_v|$ is omitted. Thus, the objective values of the old and the new model differ by this value.

However, when solving problems with D-Wave's annealing machines, we cannot choose the interaction graph G arbitrarily. It needs to correspond to the currently operating hardware graph, such as the *Chimera graph*. Only if G is a subgraph of the hardware graph, we can directly solve the ISING PROBLEM with the D-Wave annealer (with some probability) by setting surplus parameters to 0.

2.3 Chimera Graph

In this section, we introduce the specific hardware graph of D-Wave's quantum annealer, the *Chimera graph*. It was first described in [42] and released in 2011 with the first machine. The Chimera graph is derived from the structure of the overlapping superconducting loops forming the qubits and represents their connectivity. Due to the arrangement in a checkerboard pattern, it is easily extendable. Thus, the term rather refers to a family of graphs depending on a given size. We define:

Definition 2.3. Let \mathbf{C}_{cr} be the Chimera graph with the number of columns $c \in \mathbb{N}$ and the number of rows $r \in \mathbb{N}$ of *unit cells* of complete bipartite subgraphs with 4 vertices in each partition. The $K_{4,4}$ -unit cells are arranged in a grid pattern and connected as shown in Figure 2.1. In accordance with the figure, we also add the attribute of the *orientation* to the vertices: We say a vertex is *oriented horizontally* if it lies in the horizontal partition of a unit cell, is thus connected vertically to vertices in other unit cells above and/or below, *oriented vertically* otherwise. In short, we use the term *horizontal*, respectively, *vertical vertex*. Therefore, let $V(\mathbf{C}_{cr}) := \mathbf{V}_{\text{hori}}(\mathbf{C}_{cr}) \cup \mathbf{V}_{\text{vert}}(\mathbf{C}_{cr})$ with the corresponding sets of vertices.

Let \mathbf{C}_{∞} be the Chimera graph with an infinite number of unit cells in all 4 directions. This means we also allow negative row and column unit cell coordinates. Thus, there is no boundary and all vertices have a degree of 6. We call this graph *infinite Chimera graph* in the following.

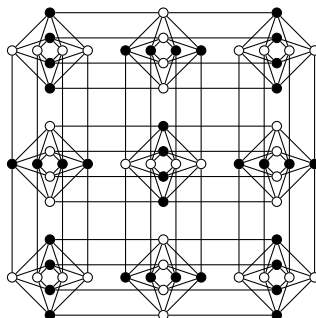


Figure 2.1: Bipartition in the Chimera graph $C_{3,3}$

Although all hardware released until now is symmetric, which means we have $r = c$, these values can vary in theory, as well as the number of vertices in the unit cell partitions. We refer to the latter as the *depth* of the Chimera graph. In this work, we deal with the ‘standard’ depth of 4.

The Chimera graph is bipartite itself, as it can be seen by the vertex coloring in Figure 2.1. Wherever necessary, we use the terms *ideal* or *non-broken* Chimera graphs for C_{rc} or C_∞ to clearly distinguish them from the following:

Definition 2.4. We call a Chimera graph *broken* if it is a vertex-induced subgraph of an ideal Chimera graph, that is, $C_\infty[W]$ for some $W \subseteq V(C_\infty)$.

Let \mathcal{C} denote the set of all finite broken Chimera graphs, which means derived from $C_{rc} \subset C_\infty$ with $c, r < \infty$.

In this work, we only refer to vertex-induced subgraphs because stand-alone broken edges are very rare in current hardware. Even if such an edge appeared in an operating machine after the calibration, we could simply mark one of the incident vertices as broken and thus continue with the same derivations as follows.

To provide insight, a currently operating D-Wave machine has 16 times 16 unit cells in the Chimera hardware structure. It is referred to as solver DW_2000Q_6 in the current calibration status. From the 2048 vertices implemented in the hardware, seven vertices are considered to be broken. With them, 40 edges are removed and only two additional edges are also broken [14].

Just recently, a new hardware architecture was released operating in parallel to the Chimera machines. The corresponding graph is called the *Pegasus graph*. It is derived from the Chimera structure by stretching the superconducting loops. As this is done asymmetrically, the Pegasus graph has a much more complicated structure but realizes a larger connectivity. In particular, the unit cell connectivity is not limited anymore to unit cells neighbored in the grid pattern but rather connects them diagonally in different ways. This however is currently only achieved at the expense of a more than six times larger ratio of broken qubits.

In this work, we mainly focus on the Chimera architecture, but we are confident that most of our results can be transferred because the Pegasus graph is still closely related to the Chimera graph. This can also be seen in the example of Section 3.2.5, where we use the subgraph relation between both graphs to prove the NP-completeness of the embedding problem restricted to broken Pegasus graphs based on the equivalent result for the Chimera graph.

3 Minor Embedding

Once the Ising model for a specific application, which shall be solved on a D-Wave machine, is formulated, we are given the interaction graph corresponding to the quadratic terms. This problem graph does in most cases not have any relation to the Chimera or Pegasus hardware graphs, if it is not explicitly customized to fit them. Therefore, the first step to calculate on the annealing machines is to solve the embedding problem, which is investigated in more detail in this chapter.

The work presented in this chapter is already published in two articles. The proof of the NP-completeness of the embedding problem when restricting to the specific hardware graphs of Section 3.2 is presented in [36] and was developed in collaboration with Annette Lutz. The examination of the special case of a complete problem graph of Section 3.3 appeared in the Journal ‘Quantum Information Processing’ [37] and was produced in joint work with Lukas Schürmann and Dr. Tobias Stollenwerk. Both publications are adopted with only slight revisions due to a uniform style of this work and the extraction of common preliminaries. With the latter, we start in the following Section 3.1.

3.1 Minors and Embeddings

When dealing with the D-Wave annealing machines, we always have the discrepancy between the problem graph and the actually realized hardware graph. Usually, we refer to both as G and H , respectively. In this section, we connect the two concepts of minors and embeddings and transfer them into the context of quantum annealing. As the embedding is essential to run experiments on the D-Wave quantum annealing machine, it is studied broadly in this context. Therefore, we summarize the existing approaches at the end of this section.

3.1.1 Basic Definitions

Graph minors are a basic concept in graph theory. We refer to [18] and [34] for more details. In the following, we introduce graph minors and their relation to the embedding as well as some hardness results in a more descriptive way.

Although several different definitions for graph minors exist, most of them are equivalent. We use the following one from [34]:

Definition 3.1. Given two arbitrary graphs G and H , G is called a *minor* of H if G is isomorphic to a graph that can be formed from H by a series of the following operations: edge contraction, edge or vertex deletion.

The term 'edge contraction' means here that the endpoints of a single edge are replaced by a single new vertex whose neighbors are formed by the union of the neighbors of these endpoints. By this no multiple edges or loops appear and we always keep the resulting graphs simple. Furthermore, note that, if we reasonably assume that no vertex is deleted that is formed by an edge contraction, the ordering of the operations is arbitrary. It is also easy to see that the minor relation is transitive, which means a minor of a minor of some graph H is itself a minor of H .

Several results concerning minor relations exist. The most straightforward question is a classical problem in graph theory:

MINOR CONTAINMENT PROBLEM. Given two arbitrary graphs G and H , does H contain G as a minor?

It is a generalization of the SUBGRAPH HOMEOMORPHISM PROBLEM mentioned in [20]. We recall the analogously applicable proof for the following hardness result because we use the same argumentation later on:

Lemma 3.2. *The MINOR CONTAINMENT PROBLEM is NP-complete.*

Proof. The problem is in NP since the edge contractions and edge and vertex deletions that need to be applied to H provide a polynomial certificate. Moreover, it includes the NP-complete HAMILTONIAN CYCLE PROBLEM as a special case by G being a cycle graph with the same number of vertices as H . □

We briefly connect the given minor definition here with a concept that is more widely used in the quantum annealing context. There graph minors usually appear in terms of an *embedding*: To overcome the very restricted hardware architecture and simulate an arbitrary problem connectivity, several hardware vertices need to be combined to form a logical vertex. This is, e.g., described in [13]. In [18], we find the more formal definition:

Definition 3.3. For two graphs G and H , an *embedding* of G in H is a map $\varphi : V(G) \rightarrow 2^{V(H)}$ fulfilling the following properties, where we use $\varphi_v := \varphi(v)$ for $v \in V(G)$ for shortness:

- a) all φ_v for $v \in V(G)$ induce disjoint connected subgraphs in H , more precisely
 - we have $\varphi_v \cap \varphi_w = \emptyset$ for all $v \neq w \in V(G)$ and
 - $H[\varphi_v]$ is connected for all $v \in V(G)$,
- b) for all edges $vw \in E(G)$, there exists at least one edge in H connecting the sets φ_v and φ_w , which means we have $\delta(\varphi_v, \varphi_w) \neq \emptyset$.

We call G *embeddable* into H if such an embedding function for G and H exists.

With this we can formulate an analogous question to above:

EMBEDDING PROBLEM. Given two arbitrary graphs G and H , is G embeddable into H ?

We can easily see that the MINOR CONTAINMENT PROBLEM and the EMBEDDING PROBLEM are equivalent in the following sense:

Lemma 3.4. *Given two arbitrary graphs G and H , G is a minor of H if and only if G is embeddable into H .*

Proof. We start with the implication, if G is embeddable into H , it is a minor of H . Given the map φ , the graph G can be obtained from H by successively contracting all edges in $H[\varphi(v)]$, labelling the resulting vertex with v for all $v \in V(G)$ and deleting all surplus vertices and edges that are not in $V(G)$ and $E(G)$, respectively.

Reversely, if G is a minor of H , we can always find an embedding: All vertices concerned by edge contractions that finally form a single vertex v of G are included in its embedding $\varphi(v)$, while all deleted vertices and edges can be ignored. \square

Due to this relation, also the term *minor embedding* is used and we refer to both problems equivalently by the MINOR EMBEDDING PROBLEM in the following.

3.1.2 In Quantum Annealing

In the quantum annealing context, the hardware graph H refers to a Chimera graph, while the problem graph G is derived from the specific application and its concrete Ising formulation and can therefore indeed be fully arbitrary. As an example, an embedding of the complete graph in the ideal Chimera graph is illustrated in Figure 3.1.

However, the quantum annealing machines do not realize an ideal Chimera graph but rather a subgraph, where certain vertices are unavailable due to physical effects. The locations of these broken vertices change with every recalibration of the machine. With the definitions of the former sections, we can now formulate the relevant question in quantum annealing:

BROKEN CHIMERA MINOR EMBEDDING PROBLEM. Given an arbitrary graph G and some $C \in \mathcal{C}$, is G a minor of C , i.e., is G embeddable into C ?

We show in the following work that this problem remains hard, although or rather because the Chimera graphs do have a very specific structure. Thus, the template approach of introducing an intermediate virtual hardware graph layer remains of practical relevance to exploit possible advantages of the annealing machines. The most general template is the complete graph, which leads to the question:

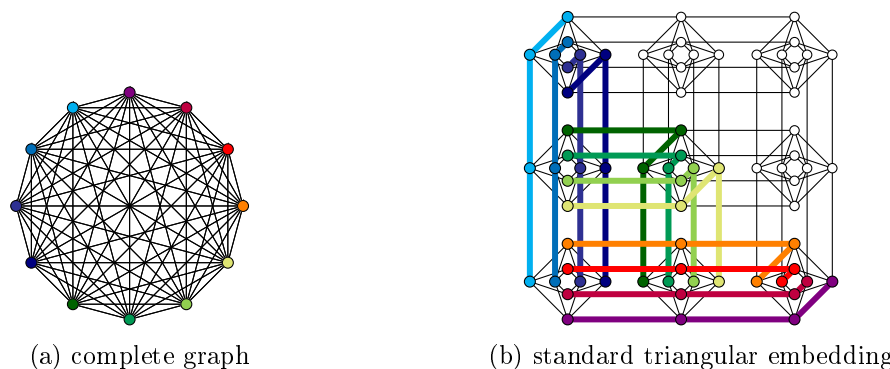


Figure 3.1: Standard triangular complete graph embedding in the ideal Chimera graph, where each color represents a single logical vertex in the complete graph

LARGEST COMPLETE GRAPH EMBEDDING PROBLEM. Given $C \in \mathcal{C}$, find the maximal $n \in \mathbb{N}$ such that K_n can be embedded into C .

It is easy to see that this is an optimization problem that reduces to the BROKEN CHIMERA MINOR EMBEDDING PROBLEM: By enumerating all complete graphs according to their size, we can check for each of them successively whether it is a minor of the given Chimera graph. If we have shown that $K_{\tilde{n}}$ is not a minor of C for some \tilde{n} , we also know that K_n is not a minor of C for $n > \tilde{n}$ because $K_{\tilde{n}}$ is a subgraph of K_n . The maximal choice of n in this procedure is bounded from above by the size of Chimera graph.

The LARGEST COMPLETE GRAPH EMBEDDING PROBLEM is also known as the search for the *largest clique minor* [7], as the complete graph can be supplemented by the vertices which are not used for the embedding of the complete graph, forming a larger graph. The largest clique of this minor corresponds to the maximal embeddable complete graph.

3.1.3 Related Work

Graph minors have been a research topic of high interest even before the D-Wave machine was released. Particularly, the work of Robertson and Seymour has mainly influenced the developments in this area. In their extensive research around such graph relations, they could show among other things that there exists a polynomial algorithm to decide whether G can be embedded in H when G is fixed to a single graph and only H is part of the input [46], although for the general problem, given two arbitrary graphs G and H , it is NP-hard to answer this question. The latter can be seen by a simple reduction, which we show in more detail in Section 3.2.1.

As the constructive approach of Robertson and Seymour requires the knowledge about what is known as the *forbidden minors* of G , their complexity result hides constants depending exponentially on the size of G . A concrete algorithm following Robertson and Seymour's idea is given for hardware graphs with fixed branchwidth in [24], which is further improved by [2], and for planar hardware graphs in [1]. Although the bounds were improved, the dependence on G is still exponential and the algorithms thus are practically infeasible for larger graphs.

The embedding problem in the quantum annealing context deals with a different case, as G is an arbitrary graph derived from the specific application, while H is fixed to be in the class of all Chimera or Pegasus graphs with broken vertices. Even though the hardware graphs are very well structured, no comparable results to Robertson and Seymour's are known in that case. In the contrary, we show in this work that the BROKEN CHIMERA MINOR EMBEDDING PROBLEM remains hard. This is stated more formally in Section 3.2.1 by Theorem 3.5.

This means the embedding problem is in general as hard as the actual problem that shall be solved on the D-Wave machine, which has some implications in practice. Usually, several possibilities to encode an arbitrary optimization problem as an Ising model exist. E.g., by introducing more vertices, the connectivity between them could be reduced. However, in view of our hardness result, we cannot expect to identify an easy to recognize set of graph properties which the chosen formulation should fulfil to be better embeddable than others or to be embeddable at all into the current hardware graph.

Up to this point, apart from problem specific approaches, like in [45], current research has mainly split up into two directions: On the one hand, the goal is to develop efficient generic heuristics that can embed as many arbitrary graphs as possible 'on-the-fly'. The first polynomial algorithm

was shown by Cai et al. in [11] and is based on finding shortest paths in the hardware graph H . As it considers both, G and H , to be arbitrary input graphs, broken vertices in a non-ideal Chimera are already taken into account. It is still the standard algorithm the package `minorminor` of the D-Wave API is based on [16]. An improvement of this algorithm is suggested in [43] and just recently compared to two new algorithms of Zbinden et al. [53], which show even better performance.

Although such heuristics work well in practice at the moment, especially for sparse input graphs, with growing hardware sizes, they will most likely not be able to scale as well. Furthermore, they have another drawback: If the heuristic fails to embed a graph, it remains unclear whether an embedding is not possible at all or whether the heuristic just could not find it. There is no guarantee that an embedding can be found or how often the heuristic needs to be repeated until we find one if it exists. We will always have to deal with the tradeoff between quality and runtime.

To overcome such a possible bottleneck, the second strategy is to insert an intermediate step in the embedding process by using a template with a precomputed fixed embedding, acting as a ‘virtual hardware’ graph. Such an intermediate layer between the actual hardware and the problem graph has a much simpler structure than the Chimera graph and shifts the expensive computation away from the user. Thus, on the one hand, the computational resources needed to calculate an embedding are decreased and, once it is found, it can be reused for the whole operational period of the machine. On the other hand, simple certificates can be formulated whether a graph is embeddable or not. The idea of precalculated templates was already introduced in [22].

A universal template is the complete graph, enabling to embed all graphs with the same or a smaller number of vertices or edges. It completely circumvents the necessity of calculating an embedding for each individual instance but rather provides it straightforwardly. Due to physical restrictions, the Chimera graph of D-Wave was designed to be sparse but nevertheless yield an efficient embedding of the complete graph [13]. The TRIAD layout, presented in [13], forms the basis for the triangle embedding structure of the complete graph in the (ideal) Chimera graph as shown in Figure 3.1(b) on page 15 for K_{12} .

Unfortunately, the shown template, as well as other existing ones, are not applicable in real hardware due to broken physical vertices. Thus, an algorithm was proposed in [32] trying to fit a related triangular scheme in the broken Chimera graph. This approach was further generalized in [7], showing even better results. As K. Boothby is one of the main contributors of the D-Wave API, we assume this algorithm is implemented in the package `minorminor`. We go into a bit more detail about their constructions when deriving our approach of [37] to solve the LARGEST COMPLETE GRAPH EMBEDDING PROBLEM in Section 3.3.1. For our algorithm solving the corresponding restricted embedding problem, we can show that the runtime is exponential only in the number of broken vertices.

Due to the very limited size of the maximal complete graph, there are various other graphs with less connectivity but a larger number of vertices considered, too. Another good template candidate is the complete bipartite graph, whose embedding is closely related to the one of the complete graph. The idea of [22] is to find the smallest number of vertices that have to be split up into the two partitions such that the resulting graph is bipartite and can thus be embedded using this template. In [47], this approach was elaborated and generalized to related partitioned graph structures.

Known minors can then be collected in a lookup table. The authors of [23] suggest to precompute all ‘maximal minors’ of the complete bipartite graph. This means an input graph is embeddable

if it is a subgraph of one of the contained minors. Another template family, for instance, is comprised of the Cartesian products of complete graphs as discussed in [52].

3.2 NP-Completeness Proof

In this section, we show the proof that the ‘minor embedding in broken Chimera and Pegasus graphs is NP-complete’, based on the correspondingly named work [36]. Our construction to prove the stated complexity result was mainly inspired by the ideas of [28]. There Itai et al. showed that the HAMILTONIAN CYCLE PROBLEM for grid graphs is NP-complete. As the HAMILTONIAN CYCLE PROBLEM reduces to the EMBEDDING PROBLEM, we adopt several of their definitions and results for the grid graphs and, in the first place, transfer them to the Chimera graphs.

For this we formulate the main question and reduction steps in Section 3.2.1 and introduce the basic grid concepts in Section 3.2.2. Afterwards, we show the construction of specific Chimera graphs in Section 3.2.3, for which we establish several results about Hamiltonian paths and cycles in Section 3.2.4. With this we can conclude the proof of the NP-completeness of the HAMILTONIAN CYCLE PROBLEM and thus of the embedding problem for Chimera graphs. The extension to Pegasus graphs is then shown in Section 3.2.5.

3.2.1 Reduction of the Hamiltonian Cycle Problem

In this section, we show how the BROKEN CHIMERA MINOR EMBEDDING PROBLEM reduces to the HAMILTONIAN CYCLE PROBLEM to adopt the known complexity results for this problem. Note that, until now, there is no result about the complexity status of the related question whether a given graph is actually itself a broken Chimera graph, more formally

BROKEN CHIMERA RECOGNITION PROBLEM. Given an arbitrary graph G , do we have $G \in \mathcal{C}$?

This means to check whether G is a vertex-induced subgraph of an ideal Chimera graph. The number of rows and the number of columns of unit cells of a corresponding Chimera graph can be bounded by the number of vertices in G . Clearly, this problem is not harder than the arbitrary INDUCED SUBGRAPH ISOMORPHISM PROBLEM, thus is in NP, but it remains open whether it is easier. We do not examine this problem further because, by the practical conditions, we know we are given a broken Chimera graph.

We proceed with the main complexity result, which we prove in the course of this section:

Theorem 3.5. *The BROKEN CHIMERA MINOR EMBEDDING PROBLEM is NP-complete.*

As the BROKEN CHIMERA MINOR EMBEDDING PROBLEM is a special case of the MINOR CONTAINMENT PROBLEM, it is in NP. But does the restriction to Chimera graphs as hardware graphs result in better solvability?

The HAMILTONIAN CYCLE PROBLEM was shown not to be hard for the ideal Chimera graph [35]. Thus, the argument in the proof of Lemma 3.2 does not apply for those graphs. However, the presented construction of a Hamiltonian cycle cannot be transferred easily if the Chimera graph is broken. In fact, we show in general

Lemma 3.6. *The HAMILTONIAN CYCLE PROBLEM for graphs $C \in \mathcal{C}$ is NP-complete.*

By this Theorem 3.5 is proven straightforwardly with the same arguments as for the general MINOR CONTAINMENT PROBLEM in the proof of Lemma 3.2. To prove in turn Lemma 3.6, we use a special set of graphs \mathcal{B} where we already know that

Lemma 3.7 ([28] Lemma 2.1.). *The HAMILTONIAN CYCLE PROBLEM for graphs $B \in \mathcal{B}$ is NP-complete.*

Here, \mathcal{B} is the set of all planar bipartite graphs with maximum degree 3. Note that two necessary conditions for the existence of a Hamiltonian cycle are an equal number of vertices in each partition and that no vertex has a degree of 1. Thus, we further restrict \mathcal{B} to such graphs in the following.

We show how to construct a broken Chimera graph, denoted by $C(B)$, from a graph $B \in \mathcal{B}$ such that it fulfils the following two properties:

Lemma 3.8. *For all $B \in \mathcal{B}$, $C(B)$ can be constructed in polynomial time.*

Lemma 3.9. *For all $B \in \mathcal{B}$, $C(B)$ has a Hamiltonian cycle if and only if there exists a Hamiltonian cycle in B .*

This means that the HAMILTONIAN CYCLE PROBLEM is hard for the constructed broken Chimera graphs with a size polynomial in the size of the graph B and it thus is hard for $C \in \mathcal{C}$ in general, holding Lemma 3.6. We have summarized the reduction steps in Figure 3.2, where the highlighted step is the one we prove in the following course of the section.

Note that we use broken Chimera graphs as given in Definition 2.4, that is, with only vertices being broken. By additionally allowing edges to be broken, the construction in Section 3.2.3 still works and could even be simplified while it would require less broken vertices. However, the

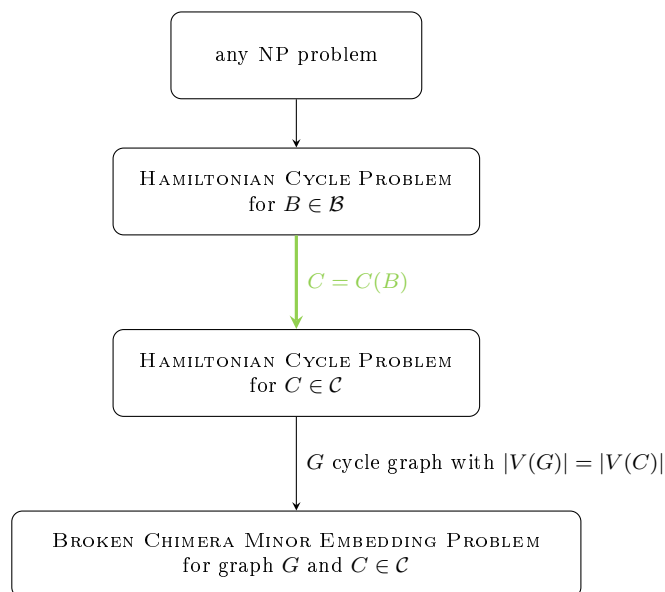


Figure 3.2: Polynomial reduction steps of the proof

probability of a broken edge is comparably small in currently operating hardware. To address the more restricted case, we focus on vertex-induced subgraphs in this work. If the hardness result holds for this subset, it also holds for all subgraphs of hardware graphs.

Furthermore, we use a *depth*, the size of the partitions in a unit cell, of $d = 4$ according to existing hardware. However, our approach can be extended to arbitrary $d \geq 4$ because a Chimera graph with depth d_1 is a vertex-induced subgraph of all Chimera graphs with depth $d_2 \geq d_1$. Thus, it is contained in the corresponding set of broken Chimera graphs.

3.2.2 Basics of Grid Graphs

Itai et al. showed an analogous result to Lemma 3.6 for grid graphs in [28]. We adopt the proof to the Chimera graph and reuse some of their constructions. For this we recall and rephrase some of their definitions and results about grid graphs in this section.

Definition 3.10. Let

$$\mathbf{G}_\infty := (\mathbb{Z}^2, \{\{ab, xy\} \subset \mathbb{Z}^2 : |a - x| + |b - y| = 1\})$$

be the *infinite (non-broken) grid graph* with vertices defined by integer *coordinates* in two-dimensional space. A *grid graph* is an arbitrary subgraph of the infinite grid graph.

A *(non-broken) rectangular (grid) graph* with one corner at coordinate $ab \in \mathbb{Z}^2$ and the opposite at $xy \in \mathbb{Z}^2$ with $xy \neq ab$ is a special grid graph and defined by

$$\mathbf{R}_{xy}^{ab} := G_\infty [\{\min\{a, x\}, \dots, \max\{a, x\}\} \times \{\min\{b, y\}, \dots, \max\{b, y\}\}],$$

where we abbreviate $R_{cr} := R_{cr}^{1,1}$ for a rectangular graph of size $c \times r$ for a number of columns $c \in \mathbb{N}$ and rows $r \in \mathbb{N}$ placed at $(1, 1)$.

Remarks:

- With the grid vertices ab, xy, ay, xb as corners, we have $R_{xy}^{ab} = R_{ab}^{xy} = R_{ay}^{xb} = R_{xb}^{ay}$.
- R_{xy}^{ab} is of size $(|a - x| + 1) \times (|b - y| + 1)$ and with $c = |a - x| + 1$ and $r = |b - y| + 1$, we have $R_{xy}^{ab} \cong R_{cr}$.

Analogously to the broken Chimera graph, we call a rectangular graph *broken* if it is a vertex-induced subgraph of a rectangular graph, the graphs [28] is dealing with. A *rectangular subgraph* denotes an arbitrary subgraph of a rectangular graph, thus a grid graph that can be bounded by a rectangular shape. We further need the following special rectangular graphs:

Definition 3.11. A *grid strip* is a rectangular graph R_{xy}^{ab} for $ab, xy \in \mathbb{Z}^2$ with

- a) $|a - x| = 1$ and $|b - y| \geq 1$ or
- b) $|b - y| = 1$ and $|a - x| \geq 1$,

thus a vertex width of 2 in one of the dimensions and arbitrary in the other. To the latter we refer as the *length* of the strip. We say a strip is *vertically oriented* in case a) and *horizontally oriented* in case b). In short, we use the term *vertical*, respectively, *horizontal strip*.

Remark: Only $R_{2,2}$ is both vertically and horizontally oriented.

The main concepts presented here are still taken from [28], but especially the following is defined slightly differently:

Definition 3.12. A *grid tentacle* is the union of a series of alternately oriented grid strips $(S_i)_{i=1,\dots,n} = (R_{x_i y_i}^{a_i b_i})_{i=1,\dots,n}$, where we have for all $i = 1, \dots, n - 1$:

- a) $V(S_i) \cap V(S_{i+1}) = \{a_{i+1}b_{i+1}, a_{i+1}y_{i+1}\}$ with $V(S_i) \cap V(S_{i+1}) \cap \{a_i y_i, x_i y_i\} \neq \emptyset$ or
- b) $V(S_i) \cap V(S_{i+1}) = \{a_{i+1}b_{i+1}, x_{i+1}b_{i+1}\}$ with $V(S_i) \cap V(S_{i+1}) \cap \{x_i b_i, x_i y_i\} \neq \emptyset$.

This means, in a tentacle, we always attach the two 'starting points' of the next strip to the side of the strip before, by which always one of the 'end points' of the latter is reused. We have the first option of the above disjunction when a vertical strip is followed by a horizontal one and the second option for the reverse case. This is shown in Figure 3.3.

Definition 3.13. Let the *parity* of a grid vertex $xy \in \mathbb{Z}^2$ given by: If $x + y \equiv 0 \pmod 2$ we call the vertex *even*, otherwise *odd*. Let the vertex sets be correspondingly denoted by the disjoint sets $V_{\text{even}}(G_\infty)$ and $V_{\text{odd}}(G_\infty)$.

It is easy to see that the even and odd vertices define a bipartition of the grid graph and with this also of grid strips and tentacles. By calling one arbitrary partition of $B \in \mathcal{B}$ even and the other odd, thus $V(B) = V_{\text{even}}(B) \cup V_{\text{odd}}(B)$, we can define analogously to [28]:

Definition 3.14. A *parity-preserving embedding* of a graph $B \in \mathcal{B}$ in the grid graph is a injective function $\psi : V(B) \rightarrow V(G_\infty)$, where

- a) the parities of v and $\psi(v)$ are the same for all $v \in V(B)$,
- b) for all $vw \in E(B)$, there exists a path from $\psi(v)$ to $\psi(w)$ in G_∞ such that the inner of all these paths do not intersect.

Note that this is a different type of embedding than in Definition 3.3 because it defines a one-to-one mapping of the vertices from the problem graph, here B , and the hardware graph, the infinite grid, while the representation of edges is generalized to paths. Although both embedding concepts are related, it is not necessary for our construction to go into details about this here.

The above definition would also be valid for arbitrary bipartite graphs, but we are especially interested in graphs $B \in \mathcal{B}$, as in [28] the authors could show that the size of the grid graph needed for a parity-preserving embedding can be bounded:

Lemma 3.15 ([28] Lemma 2.2). *For $B \in \mathcal{B}$, we can construct in polynomial time a parity-preserving embedding of B in a rectangular graph R_{nn} with $n = k |B|$ (for some constant $k \in \mathbb{N}$).*

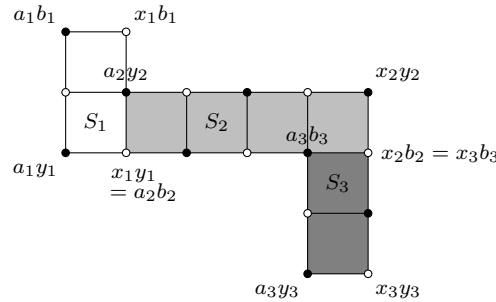


Figure 3.3: Grid tentacle example composed of three grid strips with even (white) and odd vertices (black)

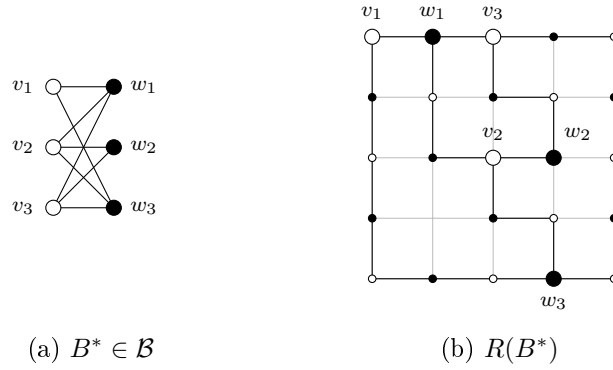


Figure 3.4: Example graph and its rectangular representation from [28] with even (white) and odd vertices (black)

For a fixed $\mathbf{B} \in \mathcal{B}$, let $\psi : V(B) \rightarrow V(R_{nn})$ be such an embedding, which we assume to be fixed in the following, too. Let $\mathbf{R}(\mathbf{B}) \subseteq R_{nn}$ denote the grid graph resulting from this embedding, where we have $\psi(V(B)) \subseteq V(R(B))$. We call $R(B)$ the *rectangular representation* of B . Note that $R(B)$ is a rectangular subgraph but not a broken rectangular graph in our sense.

An adopted example from [28] is shown in Figure 3.4. Although there might be a much simpler rectangular representation, on the one hand, it is not guaranteed to be found by the last lemma and, on the other hand, the shown example includes several different cases, which we handle in the following sections. Thus, we reuse the same example throughout the whole section.

3.2.3 Broken Chimera Graph Construction

In this section, we show the construction of $C(B)$ for an arbitrary $B \in \mathcal{B}$. To ensure that the existence of a Hamiltonian cycle is mutually induced between the original graph B and the constructed broken Chimera graph $C(B)$, we use a very restricted version of the broken Chimera graph. Although this might not be necessary for the hardness of the problem, which means we could most likely reduce the number of broken vertices, we want to keep the Hamiltonian cycle construction mostly unambitious.

We start by giving an overview about the overall concept by evolving the underlying grid graph. Afterwards, we introduce the representations of the single elements, the vertices and edges. Finally, we combine the elements to the full broken Chimera graph corresponding to B .

Underlying Rectangular Subgraph

The rectangular representation $R(B)$ is the base line for our construction. However, it is not sufficient to be used directly. In this section, we therefore evolve the full underlying grid structure of the final Chimera graph. Later the grid vertices are replaced with unit cells, where the coordinates of the grid vertices correspond to the coordinates, columns and rows, of the unit cells.

Definition 3.16. Let $\mathbf{L}(\mathbf{B})$ be the *enlarged rectangular representation* of B , a rectangular subgraph obtained from $R(B)$ by the following manipulations: We start with tripling the size of the rectangular representation $R(B)$ in each direction by replacing each edge by a straight path of

length 3. This means that an original vertex v of B , represented in $R(B)$ by $xy = \psi(v)$ for a parity-preserving embedding ψ as found by Lemma 3.15, is now represented by the coordinate $3x3y$. Due to the odd factor 3, the parity of the vertices is preserved.

Afterwards, we extend the produced graph by adding the vertices $\tilde{x}(\tilde{y} + 1)$, $(\tilde{x} + 1)\tilde{y}$ and $(\tilde{x} + 1)(\tilde{y} + 1)$ and the edges $\{\tilde{x}\tilde{y}, \tilde{x}(\tilde{y} + 1)\}$, $\{\tilde{x}\tilde{y}, (\tilde{x} + 1)\tilde{y}\}$, $\{\tilde{x}(\tilde{y} + 1), (\tilde{x} + 1)(\tilde{y} + 1)\}$ and $\{(\tilde{x} + 1)\tilde{y}, (\tilde{x} + 1)(\tilde{y} + 1)\}$ (if not present already) for each vertex $\tilde{x}\tilde{y}$ in the tripled graph. This is shown in Figure 3.5(a). However, for the odd vertices $v \in V_{\text{odd}}(B)$ with $\psi(v) = xy$ we do not use the edges $\{(3x - 1)3y, 3x3y\}$, $\{(3x + 1)3y, (3x + 1)(3y - 1)\}$, $\{(3x + 1)(3y + 1), (3x + 2)(3y + 1)\}$ and $\{3x(3y + 1), 3x(3y + 2)\}$ (if they were added before at all). After removing them, we obtain the final enlarged rectangular subgraph for our construction $L(B)$. This is illustrated in Figure 3.5(b) for a single odd vertex.

As it is shown exemplarily in Figure 3.6 on the next page, the resulting graph is now a combination of grid strips and tentacles connected by one or two edges with 'space in-between'. In the following sections, we describe the Chimera elements that replace the specific grid elements of $L(B)$. Thus, we specify:

Definition 3.17. Let $S_v := R_{3x3y}^{(3x+1)(3y+1)}$ denote the 2×2 grid strip subgraph that represents the original vertex $v \in V(B)$ in $L(B)$ for $\psi(v) = xy$. Let T_e be the grid tentacle between S_v and S_w representing edge $e = vw \in E(B)$ in $L(B)$. W.l.o.g. for v being even and w being odd, T_e is connected to S_v by two edges and to S_w by a single edge.

Concerning the overall graph, it is easy to see that

Corollary 3.18. For all $B \in \mathcal{B}$, $L(B)$ can be constructed in polynomial time.

Proof. Because the number of manipulations is bounded by the number of vertices and the number of edges of $R(B)$, the claim follows directly from Lemma 3.15. \square

Remark: The enlarged rectangular representation $L(B)$ is a subgraph of a rectangular graph of size at most $(3n - 1) \times (3n - 1)$ for constant n from Lemma 3.15.

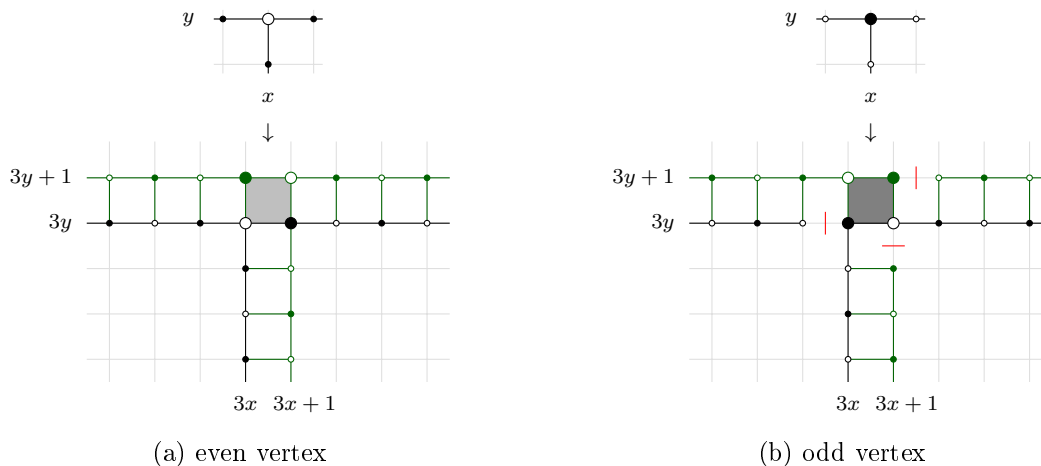


Figure 3.5: Manipulation of grid vertices and incident edges of the rectangular representation to obtain the tripled rectangular representation (black) with its extension (green) illustrated in the lower part

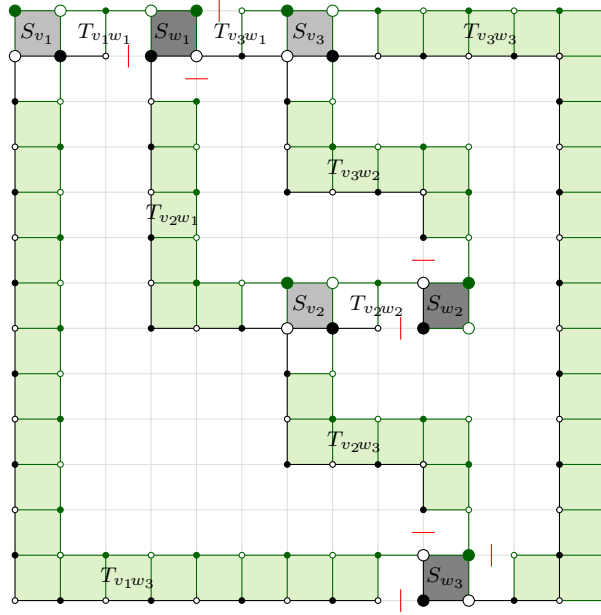


Figure 3.6: Tripled (black) and extended (green) rectangular representation forming the enlarged rectangular representation $L(B^*)$ corresponding to the example of Figure 3.4 with the sets of 2×2 vertices (underlaid in gray) corresponding to the original even (light gray) and odd vertices (dark gray) and the connecting tentacles (underlaid in light green)

Vertices as Broken Chimera Subgraphs

In this section, we explain the broken Chimera graphs that represent the single vertices. Remember each vertex $v \in V(B)$ is represented in $L(B)$ by the grid strip $S_v = R_{3x \ 3y}^{(3x+1)(3y+1)}$ with $\psi(v) = xy$ for the parity-preserving embedding ψ . Now we replace the corresponding 2×2 grid vertices by a specific broken Chimera graph of size 2×2 as illustrated in black in Figure 3.7.

In a grid graph, each vertex can have at most 4 neighbors. We restrict our construction to graphs $B \in \mathcal{B}$, where each vertex has a degree of 2 or 3. No matter where the unused edge in the rectangular representation $R(B)$ is placed, the incident edges of a vertex with degree 3 form a T-like structure, which is possibly simply rotated. All vertices of degree 2 can be represented by just dropping one of the surplus connections.

The same principle is kept for the representation of v in $L(B)$ and also in our broken Chimera graph: The blue edges added in Figure 3.7, ℓ_1 , ℓ_2 and ℓ_3 , analogously form a 'T'. Although those edges do not exist in a Chimera graph as shown, they illustrate the later connection to the Chimera elements representing the original edges. Thus, we define

Definition 3.19. Let $\tilde{\mathcal{C}}_{2,2} \in \mathcal{C}$ be the broken Chimera graph with the vertex set, and thus the connectivity, as shown in black in Figure 3.7. For $v \in V(B)$, let $\tilde{\mathcal{C}}(S_v) \subset C_\infty$ be the subgraph isomorphic to $\tilde{\mathcal{C}}_{2,2}$, possibly rotated according to the representation of the incident edges in $R(B)$ and placed at coordinate $3x \ 3y$ (with the unit cell in the lower left corner after rotation), that represents v .

As it can be seen in the definition, we use the same broken Chimera graph for all vertices in $V(B)$ regardless of their parity. The distinction of even and odd vertices only becomes apparent when

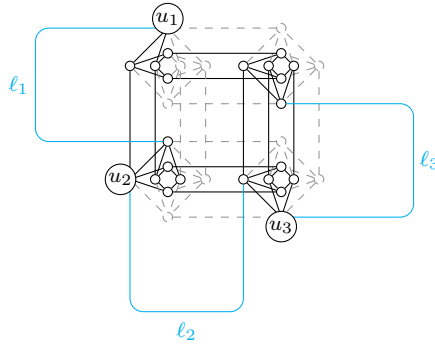


Figure 3.7: Broken Chimera graph representing a vertex with broken vertices (gray dashed) and additional edges (blue)

connecting the vertex elements with the edge elements as already indicated by the construction of $L(B)$. We go into detail about the differences in the next sections.

However, note that we did not color the Chimera subgraph's partitions in Figure 3.7 by intention. Their assignment changes depending on the parity of the grid coordinates. Furthermore, the vertices u_1 , u_2 and u_3 are labelled because they mark the main entry points of the Hamiltonian cycle constructed later on.

Edges as Chimera Tentacles

In contrast to the vertices, the construction of the edges is more straightforward because nearly all vertices in the grid tentacles of $L(B)$ are replaced by non-broken Chimera unit cells. With the grid definitions of [28] from the previous section, we define analogously

Definition 3.20. For T being a grid tentacle, let the corresponding *Chimera tentacle* be

$$\mathbf{C}(T) := C_\infty \left[\bigcup_{xy \in V(T)} V^{xy} \right],$$

where V^{xy} is the set of vertices in the unit cell at the coordinates $xy \in \mathbb{Z}^2$ in the infinite Chimera graph.

An example of a Chimera tentacle is illustrated in Figure 3.8(a). As a strip is also a tentacle with just one element, the same definition holds for *Chimera strips*.

As mentioned in the section before, the distinction between odd and even vertices is done by the way the tentacles are attached to the $C_{2,2}$ -subgraphs representing the vertices. The necessity of this distinction becomes clear in the next section when we assemble the Hamiltonian cycle in the constructed Chimera graph. According to the enlarged rectangular representation $L(B)$, we restrict odd vertices to just a single edge in each direction, while even vertices get two edges. As we focus on vertex-induced subgraphs of the Chimera graph, this is realized by additional broken vertices in the joined tentacles leading to the removal of edges. Thus, let us define slightly different Chimera tentacles to represent the edges:

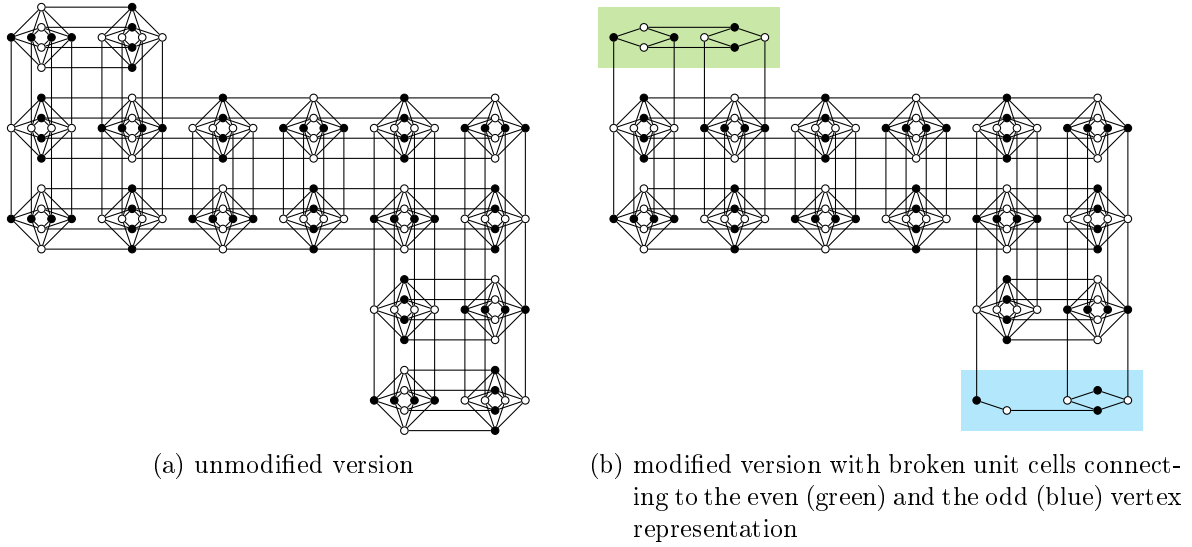


Figure 3.8: Chimera tentacles corresponding to the example in Figure 3.3

Definition 3.21. For $e = vw \in E(B)$ and, w.l.o.g., v even and w odd, let $\tilde{\mathcal{C}}(T_e)$ be the *modified Chimera tentacle* representing edge e , where the first two unit cells, connecting to S_v , are replaced by the element highlighted in green in the example in Figure 3.8(b) and the last two unit cells, connecting to S_w , by the one highlighted in blue.

Remarks:

- As we triple the rectangular representation, we ensure that we have a Chimera strip of length at least one between S_v and the next vertex or a possible corner in the tentacle in each direction for all vertices $v \in V(B)$. Thus, the above replacement is always possible in straight elements.
- If the tentacle T_e for some $e \in E(B)$ only consists of a single strip of length 1, we only use the element for odd vertices highlighted in blue.
- By the definitions, $C(T)$ and $\tilde{\mathcal{C}}(T)$ are broken Chimera graphs for all grid tentacles T .
- If T is finite, which means that the series of strips defining T only consists of a finite number of strips of finite lengths, $C(T)$ and $\tilde{\mathcal{C}}(T)$ are finite and we have $C(T), \tilde{\mathcal{C}}(T) \in \mathcal{C}$.

Composition of all Elements

In this section, we show how the single elements are combined to form the broken Chimera graph. In Figure 3.9, we show how the subgraph for the vertices and the tentacles are finally connected by an example of two neighboring vertices. Now we have all elements to combine the full broken Chimera graph. Thus, we can now formally define:

Definition 3.22. Let

$$\mathbf{C}(B) := C_\infty \left[\bigcup_{v \in V(B)} V(\tilde{\mathcal{C}}(S_v)) \cup \bigcup_{e \in E(B)} V(\tilde{\mathcal{C}}(T_e)) \right]$$

be the broken Chimera graph corresponding to $B \in \mathcal{B}$.

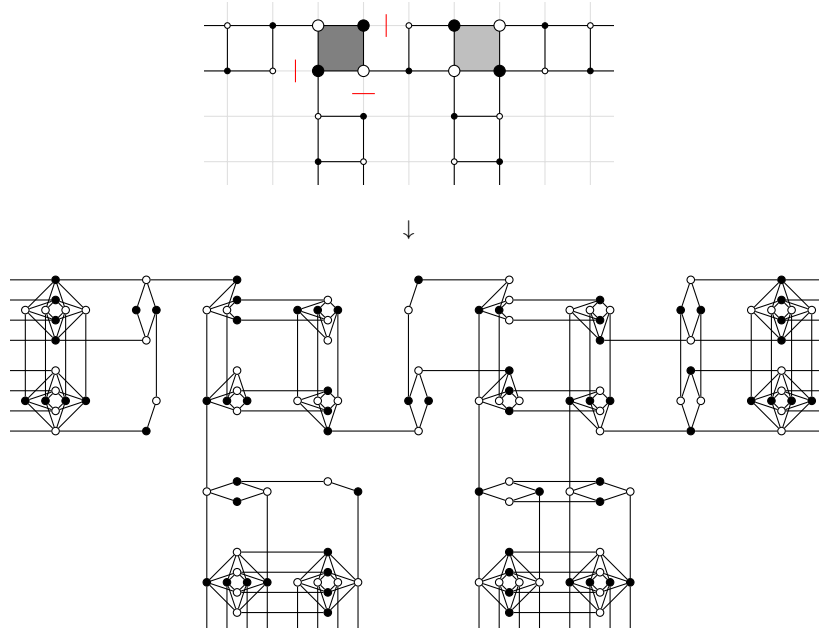


Figure 3.9: Combined Chimera graph elements derived from $L(B)$ representing neighboring odd and even vertices in $C(B)$ with three incident edges each

In other words, $C(B)$ is the broken Chimera graph derived from $L(B)$ by replacing S_v with $\tilde{C}(S_v)$ for all $v \in V(B)$ and T_e with $\tilde{C}(T_e)$ for all $e \in E(B)$ and connecting those elements with the corresponding edges. By this we can already conclude Lemma 3.8 with

Proof of Lemma 3.8. Analogously to Corollary 3.18, the replacements of grid vertices and paths of $L(B)$ by Chimera elements can be done in linear time. \square

Remember, this also means that $C(B)$ is finite (if B is finite) and we have $C(B) \in \mathcal{C}$.

3.2.4 Hamiltonicity

In this section, we establish some results about Hamiltonian paths and cycles, first in the single elements and finally in the full constructed broken Chimera graph $C(B)$. After introducing the overall concept, we evaluate Chimera tentacles first because we build on previous results on Hamiltonian paths in rectangular Chimera graphs there. After analyzing the vertex elements, we can finally combine the full Hamiltonian cycle.

Concept of Return and Cross Paths

While a Hamiltonian cycle visits every single vertex, just a subset of edges is covered. In our construction, we extend single edges to paths, which means we add further vertices. Even if the edge is not used in a Hamiltonian cycle of B , those vertices need to be covered by a Hamiltonian cycle of $C(B)$.

To handle this issue in their construction for grid graphs, the authors of [28] introduced the concept of *cross* and *return paths*. The principle is illustrated in Figure 3.10. While paths corresponding to edges used in the Hamiltonian cycle are simply passed over, thus form a cross

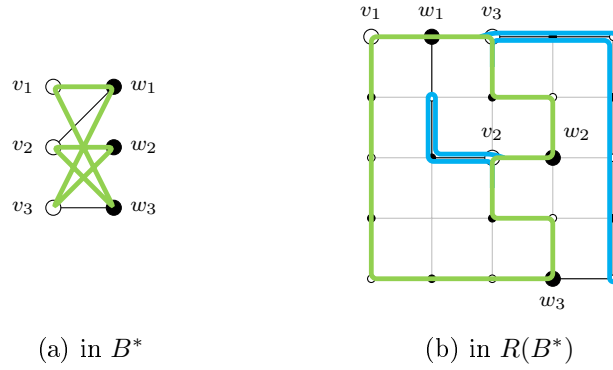


Figure 3.10: Hamiltonian cycles in the example graph and its rectangular representation of Figure 3.4 with additional loops (blue)

path, the cycle is extended by loops to cover the vertices in the remaining paths. These loops are called return paths.

Clearly, the resulting cycle in the rectangular representation is not a Hamiltonian cycle anymore because some vertices are visited twice. However, we can use this as a base line for our Chimera graph construction where the grid vertices are replaced with unit cells. As unit cells can be crossed more than once without using a vertex twice, the loops form a valid extension of the cycle finally resulting in a Hamiltonian cycle.

Due to the parity-preserving embedding, every path in the rectangular representation still connects vertices of different parity. Thus, we can decide that all the return paths start in even grid vertices representing even original vertices. In this case, the loops cross the vertices along the path until they reach the last before the odd vertex, corresponding to the original odd neighbor, and return from there.

Chimera Tentacles

In the following, we establish some results about Hamiltonian cycles and paths in the Chimera tentacles. This can be used later on to construct the desired return and cross paths in the full broken Chimera graph $C(B)$. Note that we only handle finite strips and tentacles.

The Chimera graph consists of unit cells of complete bipartite subgraphs with an equal number of vertices in the partitions in the non-broken case. Thus, paths completely covering these unit cells need to fulfil a certain condition: Each unit cell needs to be entered and left by such a path over an equal number of horizontal and vertical edges. This can be realized in different ways, for which we show some combinations in Figure 3.11. One or more vertex pairs, consisting of one horizontal and one vertical vertex each, can be traversed after each other to cross the unit cell in an L-shape. Alternatively, by splitting up these pairs, the unit cell can be crossed straight or in a U-turn. Note that the latter parts can only appear in pairs when there are no broken vertices.

For the Chimera tentacles handled in the following, we only use L-turns. Those can still be realized even if pairs of horizontal and vertical vertices are removed, like in unit cells at the beginning and the end of the modified Chimera tentacles. After establishing the general results for the non-modified Chimera tentacles, we show how to transfer them to the modified versions required for $C(B)$.

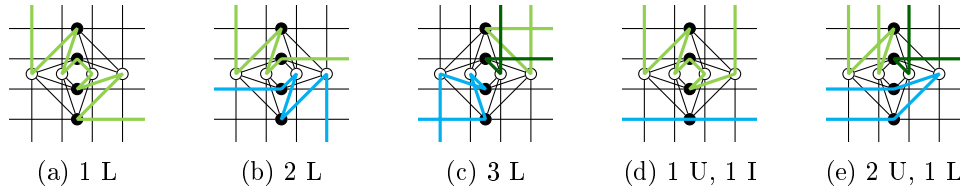


Figure 3.11: Different possibilities for parts of a Hamiltonian path to cross a unit cell, where the letters L, U and I stand for L-turn, U-turn and crossing straight, respectively

Realizing Return Paths In our Chimera construction, we need to make sure that the Chimera tentacles are suitable to realize return paths. By [35], we know every Chimera strip has a Hamiltonian cycle. There the concept of alternating 'snake paths' was shown to cover all unit cells of a rectangular shaped Chimera graph crossing them only using L-turns. Here, we briefly recall the construction and extend the statement to Chimera tentacles:

Lemma 3.23. *There exists a Hamiltonian cycle for all (finite) Chimera tentacles.*

Proof. We provide the single elements that can be combined to form a cycle covering all vertices in all unit cells of the Chimera tentacle. The cycle consists of two parallel parts along the long sides of the strips: Like in the construction of [35], both parts alternately loop to the other side of the strips by doing two L-turns. The principle is illustrated in the underlying grid tentacle in Figure 3.12(a) on the next page. The parts are finally connected in the beginning and the end of the tentacle to close the cycle.

On the Chimera level, the L-turns of the alternating parts are realized by at least one vertex pair, consisting of a horizontal and a vertical vertex, in each unit cell. The remaining vertex pairs inside the unit cell could be used in either of the parts. For symmetry reasons, we use two pairs in each straight part, however, this is not required. An example is shown in Figure 3.12(b). In the unit cells in the corners and at the ends of the tentacle, we can do the turn by simply covering all vertices, which is shown in Figures 3.12(c) and (d). In the inner unit cell of the corner, the lower left in Figure 3.12(c), the parts meet over three L-turns. This way, we can assemble the full cycle as shown in the example in Figure 3.12(e).

Finally, in Figures 3.12(b)-(d), we show the possible set of components that is sufficient, apart from rotation of the elements and permuting the ordering of horizontal or vertical vertices in a unit cell due to the symmetry, to construct a Hamiltonian cycle in a tentacle. \square

From the constructed Hamiltonian cycle, we can now easily get a Hamiltonian path that allows to cover the whole tentacle by a return path.

Corollary 3.24. *Let $C(T)$ be a Chimera tentacle derived from a grid tentacle T beginning with strip $S = R_{xy}^{ab}$. There exists a Hamiltonian path in $C(T)$*

- a) *from a horizontal vertex in V_{ab} to a horizontal vertex in V_{xb} if S is oriented vertically or*
- b) *from a vertical vertex in V_{ab} to a vertical vertex in V_{ay} if S is oriented horizontally.*

Proof. We can use the same construction as before in the proof of Lemma 3.23 just with an open straight element rather than a closing loop at the beginning of the tentacle. \square

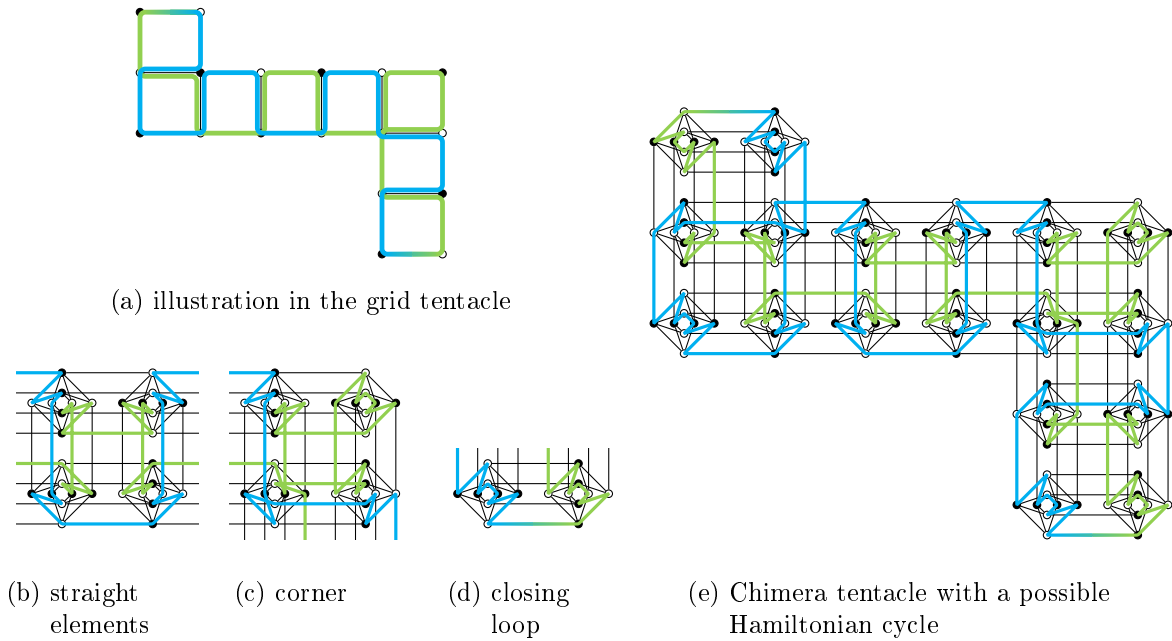


Figure 3.12: Hamiltonian cycle construction in the Chimera tentacle example with the two different alternating parts

Remark: By cutting the closing loop between the final unit cells, we can also find a Hamiltonian path from a vertical vertex in V_{ab} to a vertical vertex in V_{xb} if S is oriented vertically or from a horizontal vertex in V_{ab} to a horizontal vertex in V_{ay} if S is oriented horizontally. However, this is not necessary for our construction.

Realizing Cross Paths Similarly to the paragraph before, we show the suitability of Chimera tentacles to realize cross paths here. First of all, we establish a relation between the partitions of the Chimera graph and the parity of the grid vertices. W.l.o.g., we can specify the parity of the Chimera graph vertices in the following way: The vertices in the horizontal partition of the unit cell at coordinate $(1, 1)$ are called *even* and in the vertical partition *odd*.

Due to the alternating orientation of the partitions of the Chimera graph over the unit cells, remember Figure 2.1, the parity of the other vertices can directly be deduced from the parity of the unit cell coordinates: The above relation is repeated in every unit cell at an even grid coordinate, while it is reversed if the grid coordinate is odd. In other words, for two different unit cells with coordinates of the same parity their horizontal vertices belong to the same partition, so do the vertical vertices. Now we can simply deduce

Corollary 3.25. *Let $v \in V_{ab}$ and $w \in V_{xy}$ with $ab, xy \in \mathbb{Z}^2$ be two vertices in C_∞ . The vertices v and w are of different parity if either*

- a) *ab and xy have different parity and v and w are oriented equally or*
- b) *ab and xy have the same parity and v and w have different orientation.*

With these parity relations, we can now build the base for Hamiltonian paths from 'the beginning' to 'the end' of a Chimera tentacle, thus forming a cross path. We start with simple strips and show the following claim exemplarily for horizontal strips, but it symmetrically holds for vertical ones (by replacing xb with ay in the last sentence).

Lemma 3.26. *Let $S = R_{xy}^{ab}$ be a horizontal grid strip. There exists a Hamiltonian path from $s \in V_{ab}$ to $t \in V_{xy}$ in the Chimera strip $C(S)$ if s and t have different parity. The same holds if we replace xy with xb (the other corner point at the end of the strip).*

Proof. The partitions in a Chimera strip are of the same size due to the equal number of horizontal and vertical vertices in the full unit cells. Thus, a Hamiltonian path needs to start in one partition and end in the other. It remains to show that there always exists a Hamiltonian path in this case.

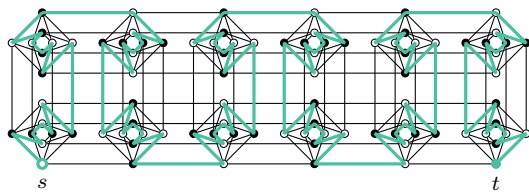
As already mentioned before, we need to enter, respectively, leave a unit cell horizontally in the same number as vertically. The easiest way to realize a Hamiltonian path respecting this is a single snake path only consisting of L-turns, analogously to one of the parts as in the proof of Lemma 3.23.

Assume the starting vertex s belongs to the vertical partition. By iteratively covering all vertices of a unit cell once entered, the path snakes along the strip as shown in Figure 3.13(a). The principle can be illustrated more simply in the grid, see Figure 3.13(b).

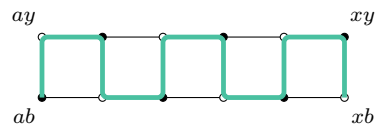
Such a snake path has the following properties:

- The snake path can be extended analogously to cover a strip of arbitrary length.
- Due to the symmetry of the unit cells, we can interchange the ordering of the horizontal and the vertical vertices along the path in a unit cell arbitrarily.
- The final unit cell is either at xy or xb depending on the length of the strip.
- We always enter the final unit cell vertically and thus end up in a vertical vertex.
- As the path also respects the bipartition of the grid graph, the final unit cell has a different parity than the first one.

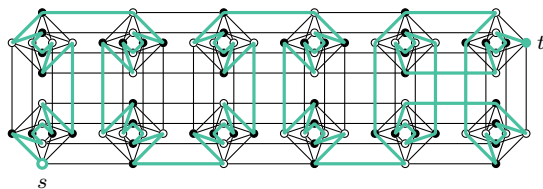
All in all, this means, if t is a vertical vertex like s , it needs to be in the final unit cell according to Corollary 3.25 and we are done.



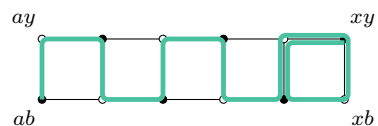
(a) snake path from a vertical vertex to a vertical vertex



(b) illustration of (a) in the grid strip



(c) snake path from a vertical vertex to a horizontal vertex



(d) illustration of (c) in the grid strip

Figure 3.13: Hamiltonian path construction in a Chimera strip example

If t is a horizontal vertex instead, it can only be in the unit cell in the other corner as in the case before. By slightly modifying the snake path using an additional loop, we can also finish the path there. This is demonstrated in Figure 3.13(c) and (d). We can observe that the final unit cell is now of the same parity as the starting one. Furthermore, it is entered horizontally, which means that the path finishes in a horizontal vertex.

In the reverse case, where s is a horizontal and t a vertical vertex we can simply use the whole construction in the reverse direction, which means we either need to rotate it, in case of $t \in V_{xy}$, or mirror it, if $t \in V_{xb}$. Note that, by both transformations, the way the path 'snakes' is inverted. Finally, if both s and t are horizontal, we need to apply the additional loop at both ends of the strip. \square

Now we can generalize the result to Chimera tentacles. Note that, with a vertex being *oriented orthogonally* to a strip it is contained in, we mean that it is oriented vertically if the strip is oriented horizontally and vice versa.

Lemma 3.27. *Let $T = \bigcup_{i=1}^n S_i$ be a grid tentacle of a series of strips $(S_i)_{i=1,\dots,n}$ for $n \in \mathbb{N}$ with $S_i = R_{x_i y_i}^{a_i b_i}$. There exists a Hamiltonian path from $s \in V_{a_1 b_1}$ to $t \in V_{x_n y_n}$ in the Chimera tentacle $C(T)$ if s and t have different parity.*

Proof. The necessity of the parity condition follows the same arguments as given in the proof of Lemma 3.26. We still need to prove the existence of a Hamiltonian path. Analogously to the proof of Lemma 3.23, we provide one sufficient set of components which can be assembled to a Hamiltonian path. Those components are derived from the paths in the single strips as shown in Lemma 3.26, which can easily be combined to form a Hamiltonian path through the whole Chimera tentacle.

We explain the concept using the underlying rectangular tentacle as illustrated in Figure 3.14(a). Assume the starting vertex s lies in a partition that is oriented orthogonally to the starting strip S_1 . Then we can use a simple snake path through the first strip, like in the proof of Lemma 3.26, finishing in one of the two 'end points'. When crossing over to the next strip, we have two cases: Either the next strip is attached to the side of the strip where the snake path ends or it is attached to the other side.

In the first case, the path can be extended along the edge that both strips share back to the already covered other starting point of the next strip. By this an inner corner is formed and the snake path can continue in the next strip. This is shown in Figure 3.14(a) in the left corner. In the latter case, we can use the additional loop as explained in the proof of Lemma 3.26 to move to the other side of the strip and continue from there, forming an outer corner, like in Figure 3.14(a) at the transition from S_2 to S_3 .

By successively adding the next strip according to the specific case, we can assemble the whole path ending up again in a vertex being oriented orthogonally to the final strip. Using the additional loop again, we can change the final vertex to be in a partition oriented equally as the strip. The remaining cases can be achieved by adding the loop in the beginning of the tentacle. However, this again switches the route of the snake path and thus the cases for all corners as well.

Because the whole construction only uses L-turns and does not enter a grid vertex representing a unit cell more than twice, it can easily be realized on the Chimera level like shown in Figure 3.14(e) with the components of Figures 3.14(b)-(d). With all rotated and reordered variants of these components we can assemble Hamiltonian paths in all Chimera tentacles. \square

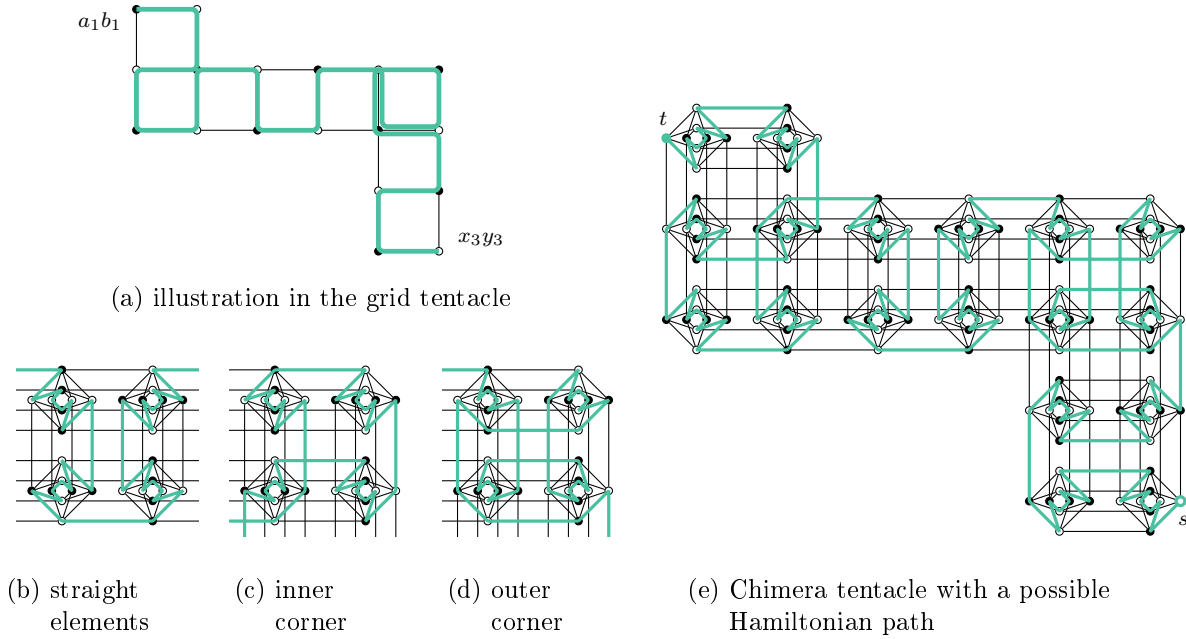


Figure 3.14: Hamiltonian path construction in the Chimera tentacle example

Modified Chimera Tentacles The previous statements about the Hamiltonicity always refer to arbitrary Chimera tentacles. However, we use the modified version in the constructed broken Chimera graph $C(B)$. But we observe the results are easily transferable:

Corollary 3.28. *Lemma 3.23 and Corollary 3.24 still hold for modified Chimera tentacles.*

Proof. In the proof of both claims, the unit cells corresponding to the start and end points are fully covered by forming an L-turn in the underlying rectangular representation. Pairs consisting of one horizontal and one vertical vertex can be removed from these closing elements until a single pair is left without disturbing the construction. \square

Unfortunately, we cannot state the same for Lemmas 3.26 and 3.27: The possibly necessary additional loops in the beginning or the end of the tentacle might require more vertices than the modified Chimera tentacles provide because the corresponding unit cells need to be entered more than once. However, by excluding these cases, we can analogously deduce the following result.

Corollary 3.29. *Let $T = \bigcup_{i=1}^n S_i$ be a grid tentacle of a series of strips $(S_i)_{i=1,\dots,n}$ for $n \in \mathbb{N}$ with $S_i = R_{x_i y_i}^{a_i b_i}$. There exists a Hamiltonian path from $s \in V_{a_1 b_1}$ to $t \in V_{x_n y_n}$ in the modified Chimera tentacle $\tilde{C}(T)$ if s and t have different parity and s and t are oriented orthogonally to the strips S_1 and S_n , respectively.*

Thus, return and cross paths can analogously be realized in modified Chimera tentacles representing edges in the constructed Chimera graph. We can now continue with connecting these paths in the corresponding vertex elements.

Vertex Elements

In contrast to the Chimera tentacles representing the edges, we have certain broken vertices placed at specific positions in $\tilde{C}_{2,2}$, respectively, in $\tilde{C}(S_v)$ for all $v \in V(B)$. Remember Figure 3.7 and Definition 3.19 of Section 3.2.3. In particular, the unit cell in the upper left corner (without rotation) has an unequal number of vertices in the different partitions. This means we cannot pair all vertices anymore, like in Figures 3.11(a)-(c), and a Hamiltonian path in $\tilde{C}(S_v)$, as a part of the overall Hamiltonian cycle, cannot cover all vertices in this unit cell by simple L-turns. It rather needs to follow a more complicated path by using an uneven number of U-turns or straight paths.

Recapturing the notation introduced with Figure 3.7, we can establish the following statement about paths in these graphs starting and finishing in the labelled vertices u_1 to u_3 .

Lemma 3.30. *There exists a Hamiltonian path in $\tilde{C}_{2,2}$ from*

- (a) u_1 to u_2 using none of the additional edges,
- (b) u_1 to u_2 using additional edge ℓ_3 (and not ℓ_1, ℓ_2),
- (c) u_1 to u_3 using none of the additional edges,
- (d) u_1 to u_3 using additional edge ℓ_2 (and not ℓ_1, ℓ_3),
- (e) u_2 to u_3 using none of the additional edges,
- (f) u_2 to u_3 using additional edge ℓ_1 (and not ℓ_2, ℓ_3).

Proof. By inspection, compare Figure 3.17 on page 37. □

Assembly

Now we have collected all the single elements that are necessary to assemble a full Hamiltonian cycle in the broken Chimera graph $C(B)$. In this section, we show that all these pieces indeed fit together.

For this let us specify the vertices forming the junctions between the elements: By \mathbf{u}_w^v , \mathbf{u}_x^v and \mathbf{u}_y^v , we denote the vertices in $\tilde{C}(S_v)$ isomorphic to u_1 , u_2 and u_3 of $\tilde{C}_{2,2}$ according to the corresponding neighboring vertices $w, x, y \in N(v)$. In other words, if the Chimera tentacle $\tilde{C}(T_e)$ for edge $e = vw \in E(B)$ replaces ℓ_i for some $i \in \{1, 2, 3\}$ in the depiction of $\tilde{C}(S_v)$ according to Figure 3.7, the vertex isomorphic to u_i is now denoted by \mathbf{u}_w^v . The vertex corresponding to u_j , $j \in \{1, 2, 3\}$, of $\tilde{C}(S_w)$ is now denoted by \mathbf{u}_w^v . Let $\mathbf{t}_w^v \in V(\tilde{C}(T_e))$ further be the unique Chimera tentacle vertex which is connected to \mathbf{u}_w^v with $t_w^v \mathbf{u}_w^v \in E(C(B))$. This notation is also illustrated in Figure 3.15.

As we do not know the trace of the Hamiltonian cycle in B in advance, and therefore also not in $C(B)$, we need to ensure that all possible ways of traversing B can equivalently be realized in $C(B)$. At first, this means wherever we have an edge between two vertices in B , we need to be able to cover the corresponding Chimera tentacle by a cross path and with this, in particular, connect the corresponding vertex representations in $C(B)$.

Although this seems trivial by the snake paths introduced before, we need to pay attention to the following: The way such a possible snake path covers a tentacle is predetermined by the single

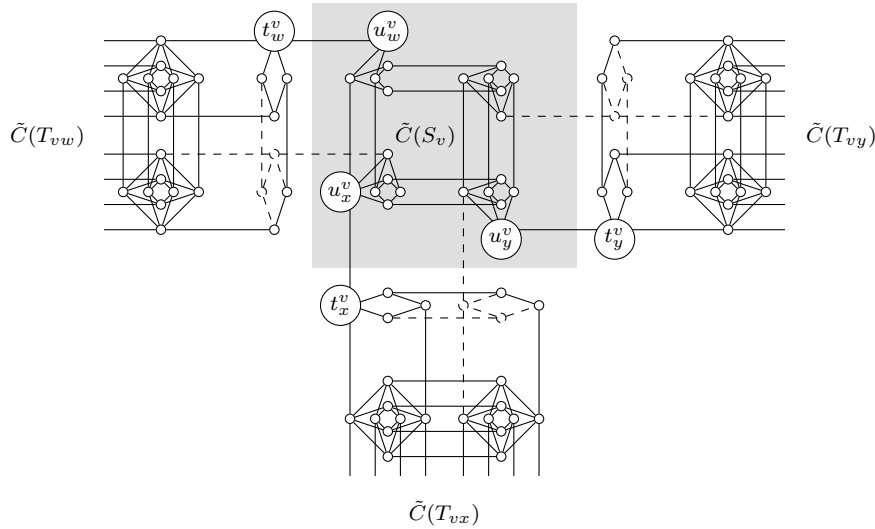


Figure 3.15: Vertices joining the vertex Chimera element $\tilde{C}(S_v)$ with the edge Chimera elements $\tilde{C}(T_{vn})$ for $n \in N(v) = \{w, x, y\}$ for v being an even vertex (including the dashed vertices and edges) or odd (excluded)

edge connecting the representation of the odd vertex with the tentacle. It is not straightforward to see that the snake path reaches the other end of the tentacle, such that we can always enter the representation of the element of the neighboring even vertex v as expected, that is, in the vertex t_w^v , from where we can enter u_w^v and continue according to Lemma 3.30. We can show that this is indeed true due to the kept properties of the parity-preserving embedding.

Lemma 3.31. *For all edges $e = vw \in E(B)$, a Hamiltonian path in $\tilde{C}(T_e)$ from t_v^w to t_w^v exists.*

Proof. In the Chimera graph, the parity of the unit cell coordinates influences which vertices of which partition of the unit cell are even or odd: In particular, for w.l.o.g. v being an even original vertex, the corresponding coordinate $3x3y$ in $L(B)$ is even and we thus observe u_n^v are even for all $n \in N(v)$. Reversely, for w as the neighbor of v being odd, all u_m^w , $m \in N(w)$, are odd, too. Note that this remains valid even if we rotate the $\tilde{C}_{2,2}$ -subgraph structure. In turn, this means that their neighboring vertices t_n^v and t_m^w are odd and even, respectively. Moreover, t_w^v and t_v^w are oriented orthogonally to their corresponding Chimera strip of $\tilde{C}(T_e)$ and therefore the conditions of Corollary 3.29 are met and we find a Hamiltonian path accordingly. \square

As the next step, we show that crossing a vertex in B , using two arbitrary edges out of three possible, can be analogously realized in $C(B)$. Because a Hamiltonian cycle always uses one edge to enter and one edge to leave a vertex, for a vertex of degree 3, we have one outgoing edge left that is not covered by a Hamiltonian cycle. The corresponding Chimera tentacle needs to be covered by a return path. Thus, we need to make sure that we are able to loop from the representation of an even vertex in all three directions. This means in turn that all odd vertices can be touched by a return path. More precisely, we show

Lemma 3.32. *For all edges $e = vw \in E(B)$, with w.l.o.g. v even and w odd, exists a Hamiltonian path in $\tilde{C}(S_v) \cup \tilde{C}(T_e)$ from u_x^v to u_y^v with $w \neq x, y \in N(v)$.*

Proof. By Lemma 3.30, we know how to traverse the vertex element from u_x^v to u_y^v analogously including the additional edge in the direction of the attached tentacle. This means that the vertex element is covered partly, left over vertex u_w^v and is entered again in the neighboring unit cell in a vertex whose partition is equally oriented to u_w^v . Equivalently, the loop in the Chimera tentacle starts and ends in the first strip of the tentacle in equally oriented vertices, more precisely, in t_w^v and the corresponding vertex in the neighboring unit cell, respectively. Thus, these vertices have different parity and there exists a Hamiltonian path between these vertices by Corollary 3.24, forming the desired return path in the Chimera tentacle. By finally covering the remaining vertices in the vertex element, we can build the whole desired Hamiltonian path. \square

Remark: In case vertex v has a degree of only 2, where w.l.o.g. y is not an element of $N(v)$, the above lemma still holds when we keep the notation u_y^v to identify the corresponding vertex. However, these cases are not relevant when constructing the overall cycle because v can only be crossed over the edges vw and vx . Thus, the element $\tilde{C}(S_v)$ can only be entered over the edges $t_w^v u_w^v$ and $t_x^v u_x^v$.

For the sake of completeness, we provide the parts of all possible pathways in the elements of a vertex of degree 3 illustrated in the grid in Figure 3.16 and in the Chimera graph in Figure 3.17. Additionally, we show in Figure 3.18 on page 38 how these parts of the Hamiltonian cycle are connected in the extracted example of Figure 3.9. With this we can now conclude our example with the illustration of the full Hamiltonian cycle in the grid in Figure 3.19.

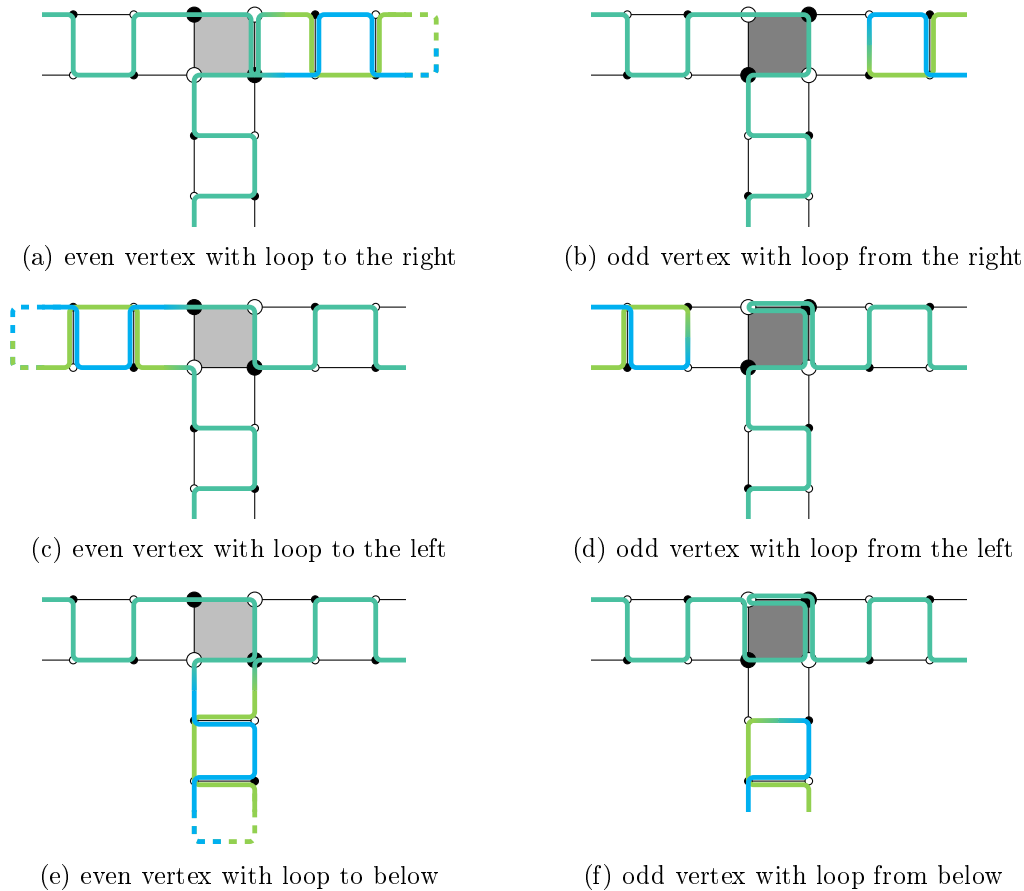
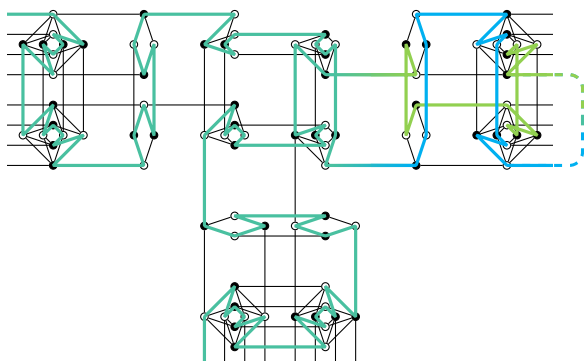
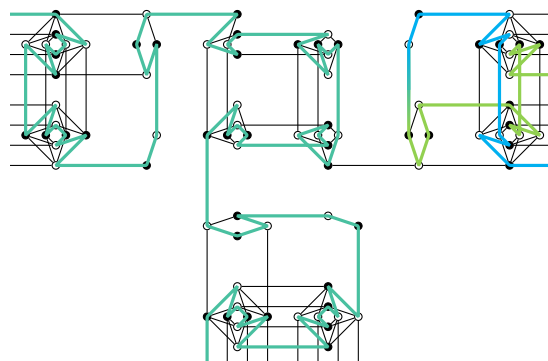


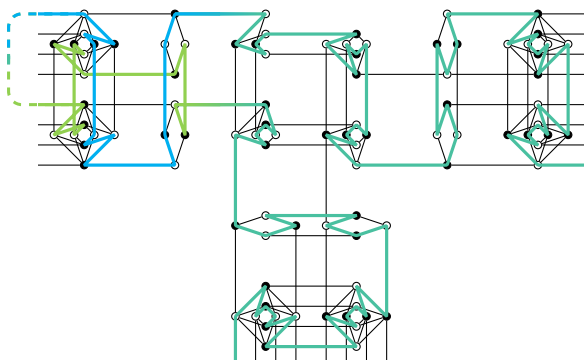
Figure 3.16: Parts of the Hamiltonian cycle through the broken Chimera graph representing a vertex with three incident edges illustrated in the underlying grid graph



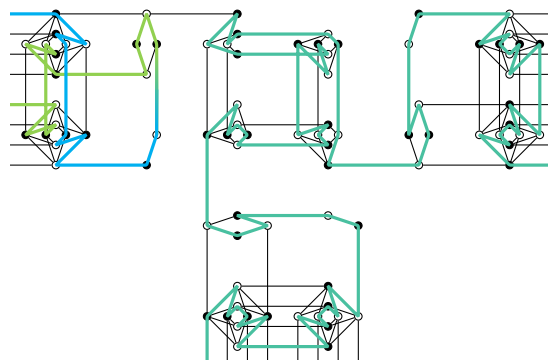
(a) even vertex with loop to the right



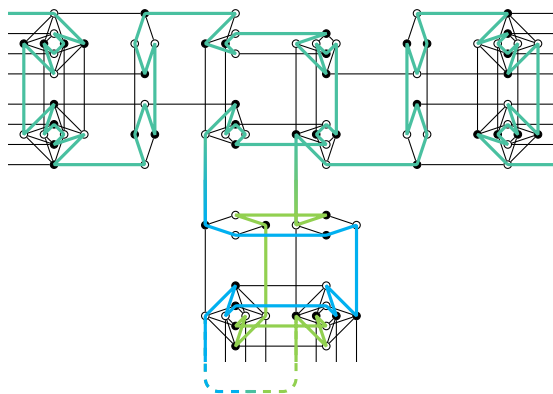
(b) odd vertex with loop from the right



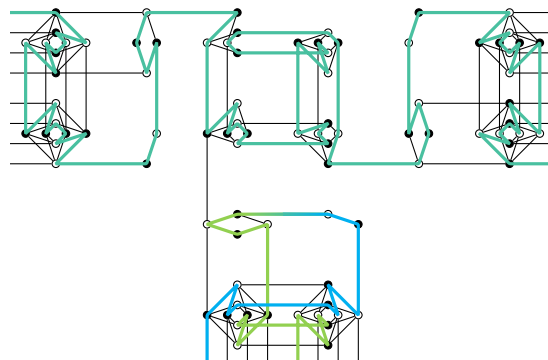
(c) even vertex with loop to the left



(d) odd vertex with loop from the left



(e) even vertex with loop to below



(f) odd vertex with loop from below

Figure 3.17: Parts of the Hamiltonian cycle through the broken Chimera graph representing a vertex with three incident edges

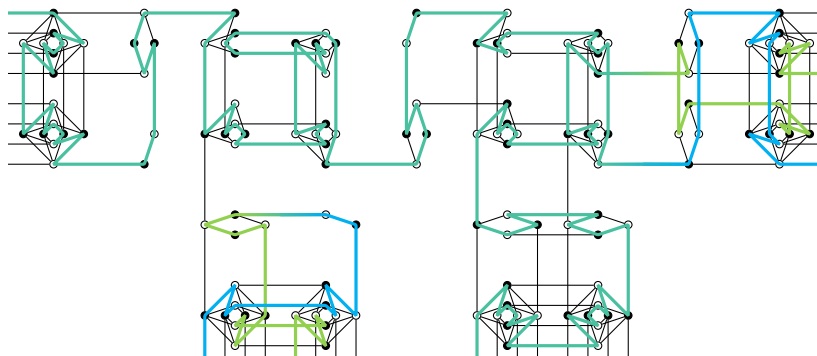


Figure 3.18: Parts of the Hamiltonian cycle in neighboring vertices of the example in Figure 3.9

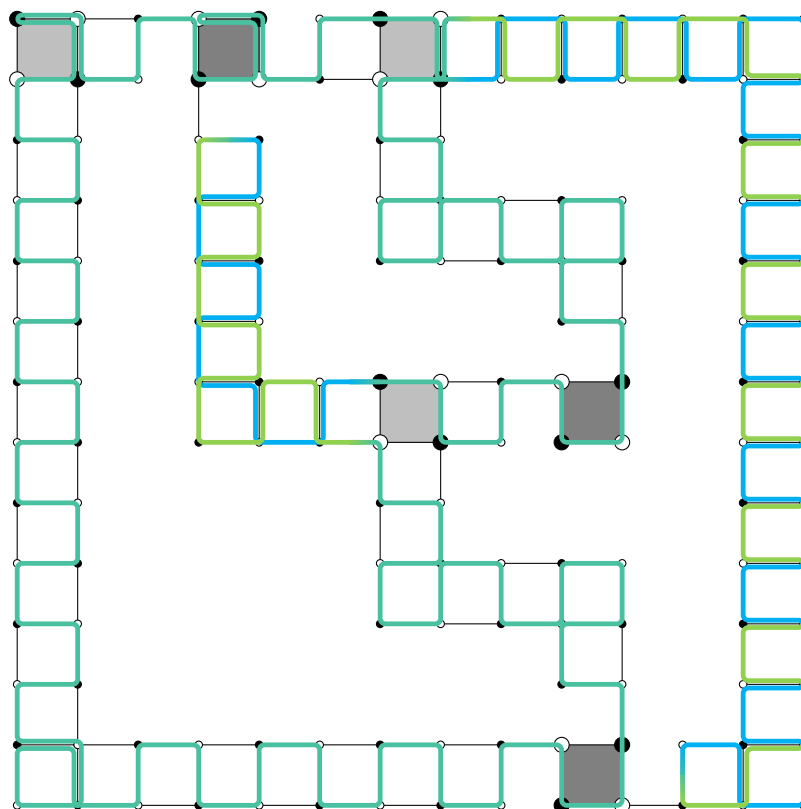


Figure 3.19: Cycle in $L(B^*)$ illustrating the Hamiltonian cycle in $C(B^*)$

Mutual Induction

In this section, we bring together the results of all the previous sections and conclude the overall complexity result with the proof of Lemma 3.9. It consists of two different parts that need to be shown for all $B \in \mathcal{B}$:

- a) If there exists a Hamiltonian cycle in B , there exists a Hamiltonian cycle in $C(B)$.
- b) If there exists a Hamiltonian cycle in $C(B)$, there exists a Hamiltonian cycle in B .

We actually prove two even stronger statements. As $C(B)$ is specifically designed to fulfil the following property, we can easily show at first:

Lemma 3.33. *A Hamiltonian cycle in $B \in \mathcal{B}$ induces a Hamiltonian cycle in $C(B)$.*

Proof. Let H be a Hamiltonian cycle in B . We have two cases for the edges in B : For all edges in the cycle $e = vw \in E(H)$, we can find a Hamiltonian path from t_w^v to t_v^w in $\tilde{C}(T_e)$ according to Lemma 3.31. For all remaining edges $e = vw \in E(B) \setminus E(H)$, let v denote the even incident vertex. As v is also covered by the Hamiltonian cycle, it needs to have two further neighbors $x, y \neq w \in N(v)$ with $vx, vy \in E(H)$. By Lemma 3.32, we can find a Hamiltonian path in $\tilde{C}(S_v) \cup \tilde{C}(T_e)$ from u_x^v to u_y^v . Thus, all vertices in the Chimera tentacles representing the edges are covered.

By the latter edge case, we have indeed handled all even vertices of degree 3 already. For the remaining vertices, $v \in V(B)$ being odd or having degree 2, we can find a Hamiltonian path from u_w^v to u_x^v equivalently to Lemma 3.30 for $w, x \in N(v)$ being the neighboring vertices where we have $vw, vx \in E(H)$. Hence, all vertices in the Chimera vertex elements are covered, too. By finally connecting all Hamiltonian path elements for the vertices and edges with the bridging Chimera edges $t_w^v u_w^v$ and $t_v^w u_v^w$ for all edges in the original Hamiltonian cycle $vw \in E(H)$, we have assembled the corresponding Hamiltonian cycle in $C(B)$. \square

For the reverse direction, we cannot assume that a possible Hamiltonian cycle in $C(B)$ exactly follows our construction. Nevertheless, we can show the analogous result to above using the following simple result about the Hamiltonicity of graph minors formed by contracting an edge of a Hamiltonian cycle:

Lemma 3.34. *Let G be an arbitrary simple graph with more than 3 vertices and Hamiltonian cycle H . Furthermore, let $e \in E(H)$ be an edge of G that is covered by the Hamiltonian cycle H , \tilde{G} be the graph that is formed by contracting edge e and \tilde{H} be the cycle in \tilde{G} that is analogously formed by contracting edge e . Then \tilde{H} is a Hamiltonian cycle of \tilde{G} .*

Proof. By contracting an edge, the cycle remains a cycle. As it still covers all vertices of \tilde{G} including the newly formed vertex from the contraction, it also remains Hamiltonian. \square

Remark: In case of G having only 3 vertices, the graph and thus also the Hamiltonian cycle is a triangle. Only in case we allow multiple edges, the graph resulting from contracting an edge still has a Hamiltonian cycle. However, by the formerly mentioned additional removal of multiple edges, the resulting simple graph collapses to a single edge and no Hamiltonian cycle exists anymore.

By *contracting along* a set of edges P forming a path, we mean in the following that each edge in P is successively contracted until just a single vertex is left. Consequently a Hamiltonian cycle

containing P in the original graph induces a Hamiltonian cycle in the graph resulting from the contraction of P . Now we can prove our final lemma

Lemma 3.35. *A Hamiltonian cycle in $C(B)$ induces a Hamiltonian cycle in $B \in \mathcal{B}$.*

Proof. Let H be a Hamiltonian cycle in $C(B)$. We show in the following that it is possible to contract $C(B)$ along paths in H , such that the resulting graph corresponds to B and the analogously contracted cycle derived from H is a Hamiltonian cycle in B according to Lemma 3.34.

For all edges $e = vw \in E(B)$ with w.l.o.g. v even and w odd, the corresponding Chimera tentacle $\tilde{C}(T_e)$ is connected to the vertex element $\tilde{C}(S_v)$ by two edges and to $\tilde{C}(S_w)$ by a single edge. Thus, H can only enter and leave the tentacle a single time. The corresponding part of the cycle forms a Hamiltonian path in the tentacle, always either from and to the even vertex representation or from the even to the odd vertex representation. How these paths are realized in detail is not of interest, but they form a kind of return or cross paths, like in our construction of a Hamiltonian cycle in $C(B)$.

We can analogously revert the steps of the construction by contracting along the Hamiltonian paths in the single elements: Each Chimera tentacle is contracted, such that only a single edge remains connecting the different vertex representations. While cross paths result in an edge covered by the resulting Hamiltonian cycle, return paths are contracted into the even vertex representation. The single edge between the representation of the odd vertex and the Chimera tentacle is kept. It is not covered by the Hamiltonian cycle in $C(B)$ and also not in the contraction. By further contracting the paths in the vertex elements until a single vertex remains, we can obtain the original graph B together with a Hamiltonian cycle in it. \square

Finally, by the last two lemmata, the proof of Lemma 3.9 follows directly and we have completed the reductions of Section 3.2.1.

3.2.5 Transfer to Pegasus

In this section, we briefly transfer our findings about the Chimera graph to the newly released hardware structure, the *Pegasus* graph. It is derived from the Chimera graph by stretching and shifting the underlying superconducting loops. By this the grid structure of unit cells is preserved, but a larger connectivity between the vertices can be realized, in particular, bridging different unit cells. As the Pegasus graph is less accessible than the Chimera, we do not go into more details here. Please see [6] for a complete formal description of the Pegasus graph P_n for $n \in \mathbb{N}$ and its different derived versions.

Here, we concentrate on the ‘standard’ Pegasus graph, which means with the default shift, as currently available in hardware by the D-Wave advantage system. We use the same term ‘broken’ as for the Chimera graph when considering a vertex-induced subgraph of a Pegasus graph. Analogously to \mathcal{C} , let \mathcal{P} be the set of all finite broken Pegasus graphs. We can now state a similar question as before, where we can show that the hardness of this question relates to the one for the Chimera graph.

BROKEN PEGASUS MINOR EMBEDDING PROBLEM. Given an arbitrary graph G and some $P \in \mathcal{P}$, is G a minor of P , i.e., is G embeddable into P ?

Figure 3.20 on the next page illustrates the relation between the Chimera and the Pegasus graph. The highlighted subgraph of the depicted Pegasus graph does nearly have the same connectivity as the Chimera graph apart from the additional edges inside the unit cells marked in red. Let \mathbf{C}_{cr}^+ be the Chimera graph derived from C_{cr} by adding these edges. This means an edge from the first to the second vertex and an edge from the third to the fourth vertex in each partition in each unit cell of C_{cr} , when enumerating the vertices along the depicted ordering. Now we can easily see from the figure that C_{cr}^+ is isomorphic to a vertex-induced subgraph of a (standard) Pegasus graph. In other words, it is a broken Pegasus graph.

By enclosing the additional edges, wherever both incident vertices are non-broken, in each partition of each unit cell in all broken Chimera elements of Section 3.2.3, we can construct analogous elements: For $\tilde{C}(S_v)$ of Definition 3.19 and $C(T_e)$ of Definition 3.20, respectively, $\tilde{C}(T_e)$ of Definition 3.21 we get $\tilde{C}^+(S_v)$ for all $v \in V(B)$ and $C^+(T_e)$, respectively, $\tilde{C}^+(T_e)$ for all $e \in E(B)$ and thus finally $\mathbf{C}^+(B)$ for $B \in \mathcal{B}$. This can still be done in polynomial time and we have $C^+(B) \in \mathcal{P}$.

Although $C^+(B)$ is not bipartite anymore, we also keep the chequered pattern of even and odd vertices. It is easy to see that all the statements, lemmas and corollaries, in Section 3.2.4 still hold when replacing all occurrences of Chimera elements with the corresponding ones from above. In particular, the proof of Lemma 3.35 remains valid because the additional edges just concern vertices inside a unit cell and the Chimera tentacles are thus still connected to the vertex elements by only three edges in total.

This way, we can deduce the following:

Lemma 3.36. *The HAMILTONIAN CYCLE PROBLEM for graphs $P \in \mathcal{P}$ is NP-complete.*

Similarly to the treatment of Chimera graphs, this implies

Theorem 3.37. *The BROKEN PEGASUS MINOR EMBEDDING PROBLEM is NP-complete.*

Note that we could replace the set of edges added between the vertices in each unit cell in $C^+(B)$ by a different one. This would not interfere with our Hamiltonian cycle construction nor with the argument for the reverse induction of a Hamiltonian cycle in the original graph. Even if we consider a Chimera graph structure but with unit cells forming a complete subgraph, the above argumentation also holds. Thus, the decisive pattern here is having a vertex-induced subgraph that consists of unit cells with a $K_{4,4}$ -subgraph being arranged and connected in a grid pattern. This means, for all new hardware architectures, if they fulfil this property, the MINOR EMBEDDING PROBLEM remains hard in general.

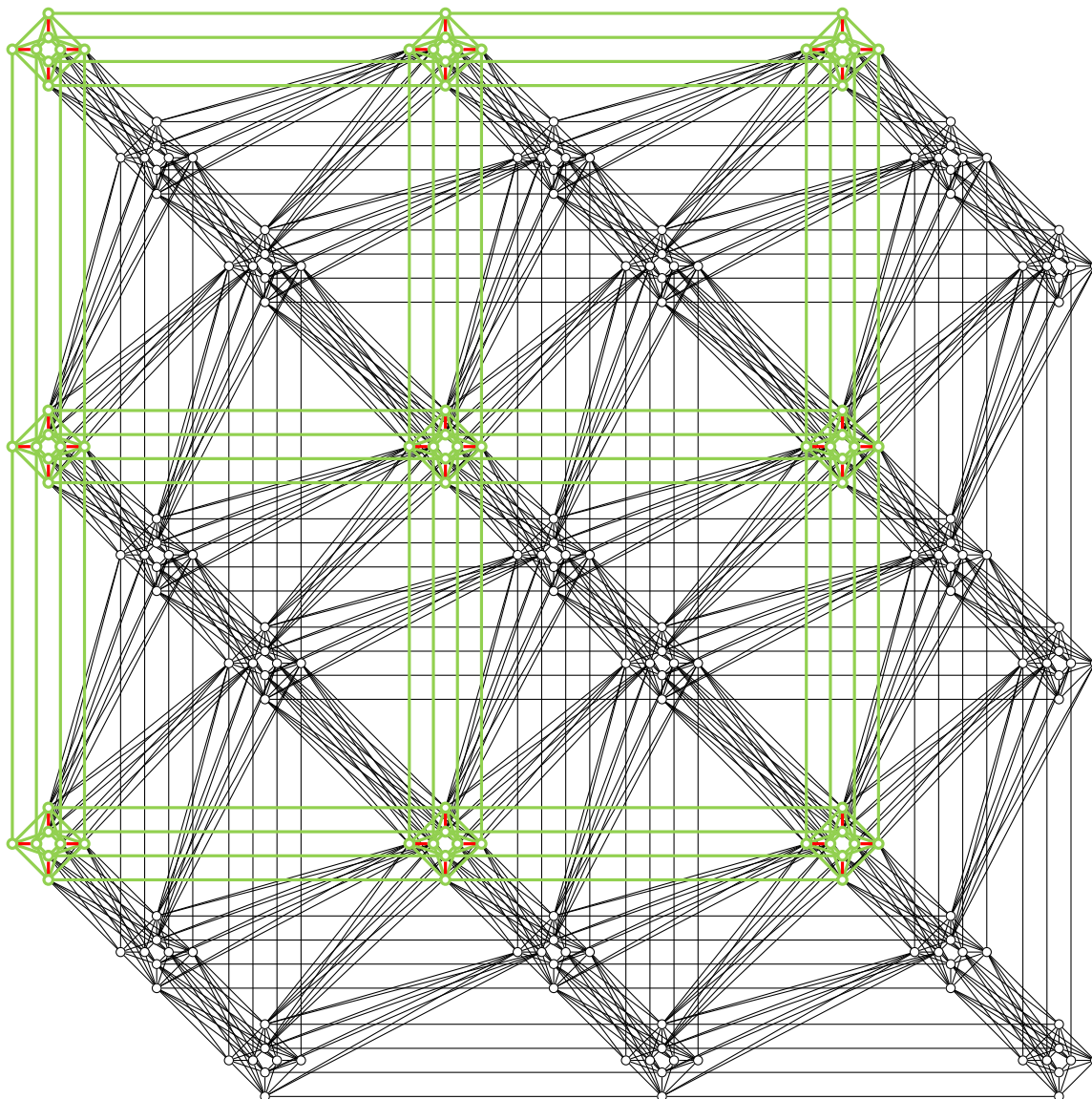


Figure 3.20: Inner of the Pegasus graph P_5 without the incomplete unit cells at the boundary

3.3 Largest Complete Graph

Although the BROKEN CHIMERA MINOR EMBEDDING PROBLEM is NP-complete for arbitrary graphs, as shown in the previous section, some special cases might yield an advantage. In this section, we investigate the complete graph as a minor serving as a universal template. By the restriction to a naturally arising embedding structure, we can formulate a restricted matching problem and show that finding the largest correspondingly embeddable complete graph in a given broken Chimera graph is fixed-parameter tractable in the number of broken vertices. The full section is based on [37].

In Section 3.3.1, we start with the explanation of the overall approach. Afterwards, we introduce a certain indexing of the Chimera graph, followed by the actual derivation of the optimization problem formulation in Section 3.3.3. At the end of that section, the complete ILP is summarized. Afterwards, the problem is analyzed theoretically in Section 3.3.4. The results of the experiments explained in Section 3.3.6 are evaluated in Section 3.3.7.

Note that, in this section, we focus on the Chimera graph only and leave the extension to the Pegasus graph, which has a larger connectivity for the same number of vertices [6], to future research. As the Pegasus graph is derived from the Chimera graph, we are confident that our results are transferable.

3.3.1 Matching Problem Approach

In the standard embedding scheme for the complete graph, remember Figure 3.1(b), the set of qubits representing a single logical vertex forms a path, which is also called a *chain* in the quantum annealing context. By extending the triangle structure, as shown in Figure 3.21(a), each of the chains becomes cross-shaped. Therefore, we call them *crosses* in the following. Additionally, each pair of crosses is now connected by two edges. Due to this redundancy, the embedding can be extended by splitting one of the crosses into its vertical and horizontal part. Thus, we can gain one additional logical vertex. According to [7], this scheme yields an embedding for the complete graph with the largest possible number of vertices. This means, in the ideal Chimera graph, the LARGEST COMPLETE GRAPH EMBEDDING PROBLEM is trivial.

To take advantage of this scheme in real hardware with broken vertices, where the shown templates are not applicable, the authors of [32] proposed an algorithm trying to extract a subgraph of the broken Chimera where the extended triangle embedding can still be applied and has as

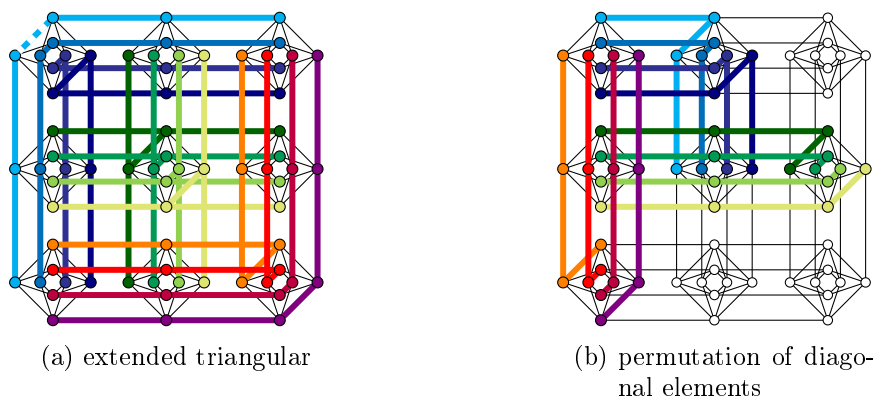


Figure 3.21: Different variants of complete graph embeddings in the ideal Chimera graph

many chains as possible. In [7], this approach is generalized by breaking up the triangle structure and placing L-shaped blocks such that all of them overlap pairwise. The principle is illustrated in Figure 3.21(b). The approach of Boothby et al. [7] to find a largest possible complete graph shows a significant advantage over [32] regarding graph sizes. In this work, we further generalize both approaches to still allow for crosses of qubits representing a single logical vertex but also open up the triangle structure.

Figure 3.22(a) shows another variant of a complete graph embedding in the ideal Chimera graph. In this construction, every row of qubits is connected to a column of qubits like in the extended triangular embedding in Figure 3.21(a). But in contrast to Figure 3.21(a), the edges connecting the horizontal and vertical cross parts of Figure 3.22(a) do not lie on the diagonal of the Chimera but are distributed over the graph. We call those edges *crossroads* in the following, forming the crosses representing a single original vertex in the embedding.

As each of the crosses occupies the full horizontal, respectively, vertical part, every row, respectively, every column of qubits belongs to a specific cross. For each row and column combination, there is a unique crossroad connecting them. Thus, such an embedding is defined by a matching of rows to columns. In turn, each matching of rows to columns defines a complete graph embedding in the ideal Chimera graph. This means there are a factorial number of possibilities to embed the complete graph.

The redundant edges connecting two crosses would again allow for one more logical vertex by spitting one of the crosses at the crossroad. However, we disregard this, as the redundancy offers another opportunity: The ends of the crosses could be cut off to make room for broken qubits as shown in Figure 3.22(b). There the remaining, shorter crosses still have an edge to every other cross and thus form a complete graph embedding. But given an arbitrary broken Chimera graph, how do we place the crosses such that this is fulfilled?

By choosing a certain edge connecting a row and a column to locate a crossroad there, the corresponding cross is well defined: Both the horizontal and the vertical part are extended until we reach the boundary of the Chimera graph or a broken qubit. To place two crossroads, we need to ensure the resulting crosses ‘meet’ each other, which means there is at least one edge connecting both. Thus, this approach can be reformulated as: How do we match rows with columns to form crossroads, like in Figure 3.22(c), such that all resulting crosses meet each other?

Clearly, the more restricted the graph the smaller is the number of suitable matchings. While just a few broken vertices might still yield a variety of complete graph embeddings, in particular, if the Chimera graph is very broken, none of the originally large number of possibilities might

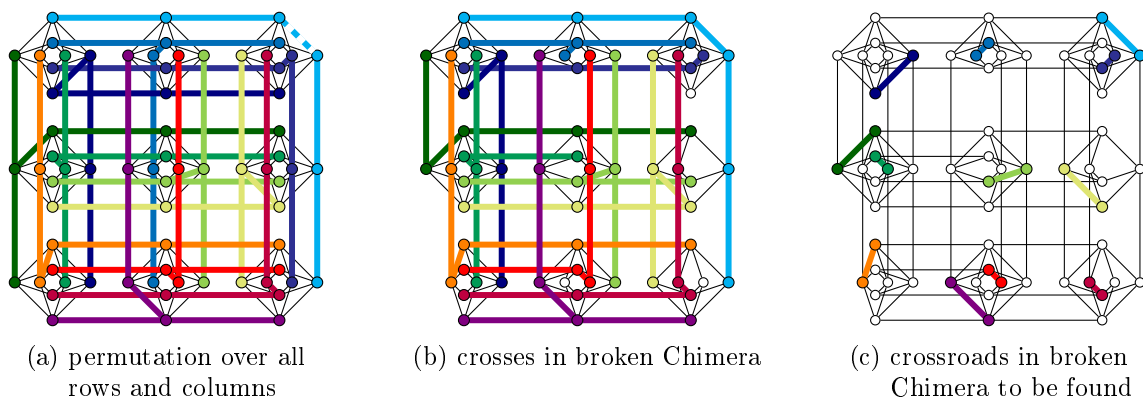


Figure 3.22: Complete graph embeddings in a broken Chimera graph with permuted crossroads

be valid anymore. Hence, we might not be able to embed the same number of vertices as in the ideal case, leading to the question: Which matching results in the largest possible complete graph? This question is an optimization problem, whose construction we show in the following sections.

For simplification of the notation, we show the construction for the standard symmetric form of the Chimera graph, like in current hardware. But this approach can be extended to arbitrary dimensions. We further concentrate on finding just a single solution rather than enumerating all possibilities of the same, optimal size because one embedding is sufficient to start calculating on the annealing machine.

The next step after the pure graph embedding in the process of obtaining an embedded Ising model is the distribution of the original problem weights over the various physical vertices. The final weight distribution depends not only on the original Ising model but also on parameters of the embedding and might influence the performance of the annealing process significantly. For instance, one of the factors that is assumed to have a relevant influence is the *chain length*, here more precise the *cross size*, that is, the number of vertices in the crosses.

As we aim for plain embeddability here, no further parameters apart from the complete graph size are part of the optimization. To get the final embedding, the crosses are extended until the boundary of the Chimera, whether this introduces redundancy in the connecting edges or not. However, exploiting this redundancy might lead to better solutions in terms of the embedding parameters: By cutting off unnecessary vertices from the end of the crosses, the cross size can be reduced. Another option is to select the embedding from the full set of equivalent optimal solutions, where, for example, the cross size is minimal. Both of the mentioned options introduce a second optimization level, which would lead far beyond the scope of this work. We keep this for future work.

3.3.2 Extension of the Chimera Graph Description

In this section, we present the Chimera hardware graph with a specific indexing of the graph vertices, being suitable for the ILP formulation of our matching version of the LARGEST COMPLETE GRAPH EMBEDDING PROBLEM, and the variable input parameters. Furthermore, we specify the sets of broken qubits according to their orientation in the Chimera graph. Note that the distinction of horizontal and vertical vertices is interchanged compared to the original publication [37] in favor of a uniform notation. Remembering Definition 2.3, we refer here to the arrangement of the vertex partitions in the depiction of the Chimera graph.

Vertex Indexing

A Chimera graph is defined by a lattice structure of complete bipartite subgraphs, which are called *unit cells*. Based on current hardware, the reference here is always the ideal symmetric Chimera graph with the number of rows and columns of unit cells given by size $\mathbf{s} \in \mathbb{N}$, which we shortly denote by $\mathbf{C}_{\mathbf{s}} := C_{ss}$. For the following construction, we assume s to be given and fixed and therefore drop the reference to C_s for shortness.

Due to the lattice structure, each vertex can be represented as a tuple of indices referring to its row and column. With $E(C_s) = E_{\text{cell}} \cup E_{\text{inter}}$ and $V(C_s) = V_{\text{hori}} \cup V_{\text{vert}}$ and the index sets $\mathbf{S} := [s]$ and $\mathbf{N} := [n]$ with $\mathbf{n} := 4s$, we have

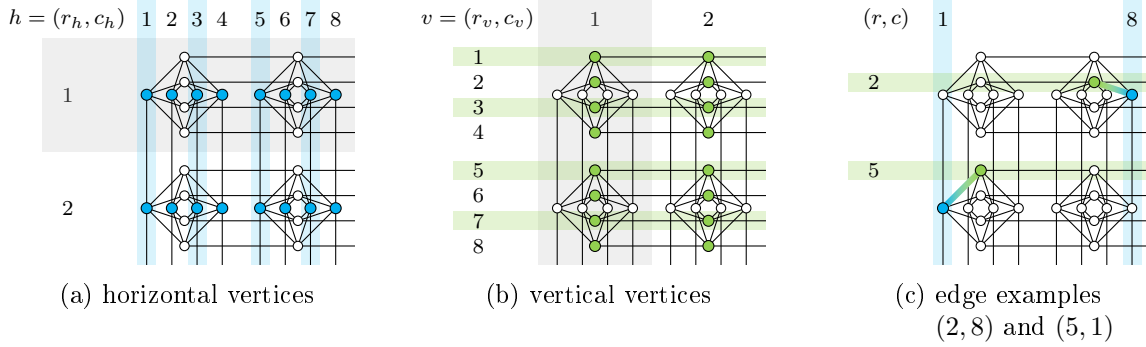


Figure 3.23: Specific indexing in the Chimera graph

- the horizontal vertices $\mathbf{V}_{\text{hori}} := S \times N$ with s rows and n columns, illustrated in blue in Figure 3.23(a),
- the vertical vertices $\mathbf{V}_{\text{vert}} := N \times S$ with n rows and s columns, illustrated in green in Figure 3.23(b),
- the inter unit cell edges $\mathbf{E}_{\text{inter}} \subset V_{\text{hori}}^2 \cup V_{\text{vert}}^2$ connecting vertices of different unit cells, which are not needed explicitly in the following and therefore are not specified here, and
- the edges inside of the single unit cells

$$\mathbf{E}_{\text{cell}} := \left\{ (h, v) = ((r_h, c_h), (r_v, c_v)) \in V_{\text{hori}} \times V_{\text{vert}} : r_h = u(r_v), c_v = u(c_h) \right\}.$$

In the latter, we use the function $\mathbf{u}: N \rightarrow S$ with $u(x) = \lceil \frac{x}{4} \rceil$, being the mapping from the *inner row/column* to the *unit cell row/column* index, in the equality constraints to ensure that the paired vertices lie in the same unit cell. Since the unit cell rows and columns are given implicitly with this function, we can use the congruent representation

$$\begin{aligned} E_{\text{cell}} &\cong \{(r_v, c_h) : r_v, c_h \in N\} \\ &= \{rc : r, c \in N\} \\ &= N^2 \end{aligned}$$

in the following. In general, we use

$$\begin{aligned} \mathbf{r}_{(\cdot)} : (x_1, x_2) &\mapsto x_1, \\ \mathbf{c}_{(\cdot)} : (x_1, x_2) &\mapsto x_2 \end{aligned}$$

for providing the row, respectively, column for a given vertex, whereas r and c (without further index) always refer to some inner row, respectively, column indices without specifying a certain corresponding vertex. Furthermore, we identify the tuple (r, c) with the non-commutative product rc for shortness to describe an inner unit cell edge. An example for the indexing of edges is shown in Figure 3.23(c).

Broken Vertices

With regard to real hardware, we consider some vertices to be unavailable. For the symmetric Chimera graph C_s with $s \in \mathbb{N}$ as described in the previous section, let $\mathbf{B}_{\text{hori}} \subset V_{\text{hori}}$ and $\mathbf{B}_{\text{vert}} \subset V_{\text{vert}}$ be the sets of different broken vertices and $B := B_{\text{hori}} \cup B_{\text{vert}}$. In our experiments,

we vary the ratio of broken vertices to the total number of vertices in an ideal Chimera graph, that is,

$$\mathbf{b} := \frac{|B|}{|V_{\text{horiz}}| + |V_{\text{vert}}|} = \frac{|B|}{8s^2}.$$

While for an ideal Chimera graph the set of possible crossroads defining the crosses in the embedding is just E_{cell} , the available combinations in a broken Chimera graph are restricted to those inner unit cell edges which do not contain a broken vertex:

$$\begin{aligned} \mathbf{A} &:= \{(h, v) \in E_{\text{cell}} : h \in V_{\text{horiz}} \setminus B_{\text{horiz}}, v \in V_{\text{vert}} \setminus B_{\text{vert}}\} \\ &\cong \{rc \in N^2 : ((u(r), c), (r, u(c))) \in E_{\text{cell}} \cap ((V_{\text{horiz}} \setminus B_{\text{horiz}}) \times (V_{\text{vert}} \setminus B_{\text{vert}}))\}. \end{aligned}$$

3.3.3 ILP Formulation

In this section, we construct an ILP for the introduced complete graph embedding problem over arbitrary input parameters s , B_{horiz} , and B_{vert} as described in the previous section.

Bipartite Matching Problem

In general, the restricted LARGEST COMPLETE GRAPH EMBEDDING PROBLEM as we consider it here is a matching problem: Which row can be matched with which column to form a crossroad in an optimal embedding following our construction rules? Thus, we call the corresponding problem the LARGEST COMPLETE GRAPH MATCHING PROBLEM, whose full ILP formulation is given later in this work, to distinguish it from the general problem formulation.

The decision which of the available combinations is taken can be encoded in binary problem variables $\mathbf{x} \in \{0, 1\}^A$ with

$$x_{rc} = \begin{cases} 1 & \text{if row } r \text{ is matched to column } c, \\ 0 & \text{otherwise.} \end{cases}$$

We say a crossroad rc is *activated* if its corresponding binary variable x_{rc} is activated in an optimal solution, which means it is set to 1. For simplification, we use $x \in \{0, 1\}^{N \times N}$ in the following with disabling all unavailable row-column pairs by presetting $x_{rc} = 0$ for all $rc \in N^2 \setminus A$, although this extends the model with additional variables.

As the goal is to match as much as possible, we want to maximize the number of activated binary variables and the objective is

$$\sum_{rc \in A} x_{rc} = \sum_{rc \in N^2} x_{rc}.$$

Our construction is based on crossroads joining full rows and columns. Therefore, only one crossroad per row and column is allowed. This can be enforced by the *matching constraints*

$$\begin{aligned} \sum_{r\tilde{c} \in A} x_{r\tilde{c}} &= \sum_{r \in N} x_{r\tilde{c}} \leq 1 \quad \forall \tilde{c} \in N, \\ \sum_{\tilde{r}c \in A} x_{\tilde{r}c} &= \sum_{c \in N} x_{\tilde{r}c} \leq 1 \quad \forall \tilde{r} \in N. \end{aligned} \tag{3.1}$$

Those types of constraints are also called *cardinality constraints* because they enforce choosing a certain number of members, here just one, out of a given set. By these restrictions, the optimal value of the objective function corresponds to the size, that is, the number of vertices, of the largest embeddable complete graph. Additionally, they confirm the upper bound on the objective function of $n = 4s$, which is the maximal size of a complete graph in C_s using our construction as explained in Section 3.3.1.

Until now, the constructed problem is just a simple MAXIMUM BIPARTITE MATCHING PROBLEM. In the following, we show further constraints that need to be added.

Mutually Exclusive Sets Constraints

If a vertical vertex is broken, it interrupts the horizontal path from the left to the right. This prevents a cross occupying this row to be extended to the boundaries of the Chimera graph. It is analogous for a broken horizontal vertex and a cross using the corresponding column. This needs to be taken into account when considering possible crossroad candidates for the embedding. Figure 3.24 depicts examples of such a situation. Due to the broken vertices, the corresponding crosses for certain pairs of crossroads might not meet. This means there do not exist any edges between the different vertices of the crosses, even if they reach the same unit cell like in Figure 3.24(b). But at least one edge is needed to provide a valid embedding of two vertices of the complete graph. Therefore, those crossroads cannot be activated together and we need to introduce further constraints enforcing the activation of only one of them.

We see that there are not only isolated pairs but clusters of crossroads all being pairwise forbidden, which means that only one of all of them can be activated. We call those clusters *mutually exclusive sets* (MES). The construction of those sets differs for certain pairs of broken vertices. We have the following cases, which are handled separately in the next paragraphs:

1. two broken vertical vertices,
2. two broken horizontal vertices,
3. two different broken vertices, one vertical and one horizontal.

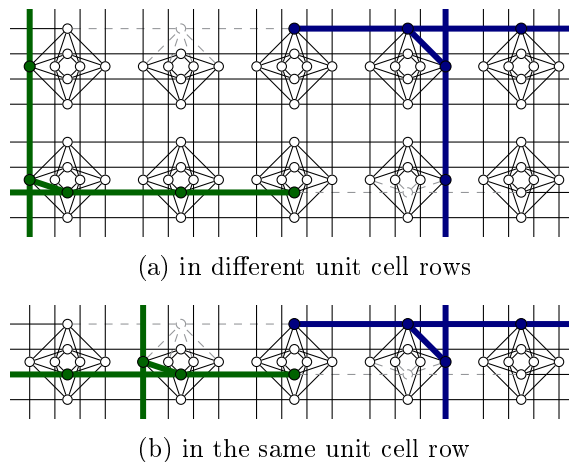


Figure 3.24: Crosses that do not meet due to broken vertices (gray dashed)

1. Let $v = (r_v, c_v) \neq w = (r_w, c_w) \in B_{\text{vert}}$ be **two broken vertical vertices**, as illustrated in Figure 3.25. Due to the horizontal interruption, the crossroads on the left and the right of the two vertices are affected. In Figure 3.25(b), both broken vertices lie in the same inner row. As the matching constraints already permit only one crossroad per row, no further constraints are necessary in this case and we can restrict to vertices with $r_v \neq r_w$, which is the case shown in Figure 3.25(a). There we have two MES, which are highlighted in different colors. Each of the blue crossroads in the right top corner cannot be activated together with the others in this corner due to the matching constraints, and they cannot be activated together with the blue in the left bottom corner because their corresponding crosses would not meet. The same holds for the green crossroads in the opposite corners.

For the definition of the MES, we need all crossroads from the left boundary until the leftmost broken vertex and all crossroads from the rightmost broken vertex until the right boundary in the two corresponding inner rows. The incident edges (light green) to the broken vertices are excluded by definition of A , respectively, set to 0, but again are included in the definition of the two sets for simplicity. Let for example v be the top left broken vertex in Figure 3.25(a) and $I_{\text{left}} \subseteq N$ describe the interval of columns on the left and $I_{\text{right}} \subseteq N$ on the right. The set of blue crossroads in the left top corner is then given by combining the row r_v with each of the columns in I_{left} . The remaining blue crossroads combine r_w with I_{right} . This results in the MES

$$(\{r_v\} \times I_{\text{left}}) \cup (\{r_w\} \times I_{\text{right}}).$$

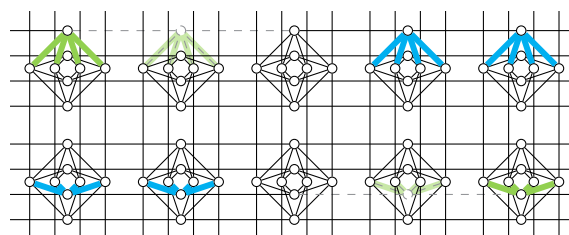
For the green crossroads, we get symmetrically

$$(\{r_w\} \times I_{\text{left}}) \cup (\{r_v\} \times I_{\text{right}}).$$

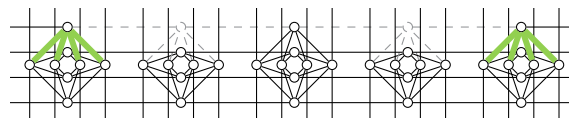
The intervals of columns, I_{left} and I_{right} , can be derived from the broken vertices' columns depending on the relational position of the vertices. To describe this more formally, for the fixed size n let

$$I(s_1, s_2) := \begin{cases} [4s_1] & \text{if } s_1 \leq s_2, \\ [4(s_1 - 1) + 1; n] = [4s_1 - 3; n] & \text{otherwise} \end{cases}$$

be the interval to or from s_1 depending on the relation to s_2 for $s_1, s_2 \in S$. The multiplication with 4 is needed for the conversion of unit cell to inner columns. The behaviour of this function is illustrated in Figure 3.26 for different relations. By the subtraction of $\frac{1}{2}$, we can circumvent



(a) two different MES due to different rows



(b) MES in same row already handled by matching constraints

Figure 3.25: Sets of mutually exclusive crossroads caused by two broken vertical vertices

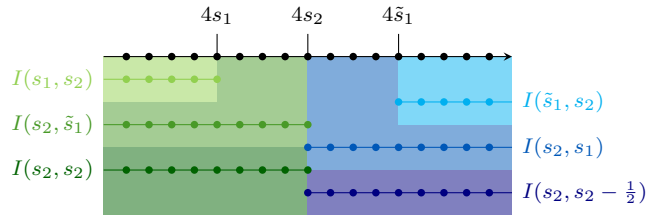


Figure 3.26: Illustration of the interval function

the fact that I only returns the left interval for identical inputs when we need the right one. The resulting sets of mutually exclusive crossroads can then be defined for each combination of v and w with

$$\begin{aligned} \mathbf{X}_1(\mathbf{v}, \mathbf{w}) &:= (\{r_v\} \times I(c_v, c_w)) \cup (\{r_w\} \times I(c_w, c_v - \frac{1}{2})), \\ \mathbf{X}_2(\mathbf{v}, \mathbf{w}) &:= (\{r_w\} \times I(c_v, c_w)) \cup (\{r_v\} \times I(c_w, c_v - \frac{1}{2})). \end{aligned} \quad (3.2)$$

Therefore, we get the cardinality constraints for $i = 1, 2$

$$\sum_{rc \in X_i(v, w)} x_{rc} \leq 1,$$

where the sum, e.g. for $i = 1$, can also be written as

$$\sum_{c \in I(c_v, c_w)} x_{r_v c} + \sum_{c \in I(c_w, c_v - \frac{1}{2})} x_{r_w c}.$$

2. Two horizontal broken vertices, with $h = (r_h, c_h) \neq k = (r_k, c_k) \in B_{\text{hori}}$, can be handled analogously to the case before by exchanging row and column: We can restrict on $c_h \neq c_k$. Taking all rows from the upper boundary to the uppermost broken vertex and all from the bottom to the lowest broken vertex results in the sets of mutually exclusive crossroads

$$\begin{aligned} \mathbf{X}_1(\mathbf{h}, \mathbf{k}) &:= (I(r_h, r_k) \times \{c_h\}) \cup (I(r_k, r_h - \frac{1}{2}) \times \{c_k\}), \\ \mathbf{X}_2(\mathbf{h}, \mathbf{k}) &:= (I(r_h, r_k) \times \{c_k\}) \cup (I(r_k, r_h - \frac{1}{2}) \times \{c_h\}). \end{aligned} \quad (3.3)$$

Therefore, we get exemplary the constraint for $i = 1$

$$\sum_{rc \in X_1(h, k)} x_{rc} = \sum_{r \in I(r_h, r_k)} x_{rc_h} + \sum_{r \in I(r_k, r_h - \frac{1}{2})} x_{rc_k} \leq 1.$$

Let us combine the MES for all combinations in

$$\begin{aligned} \mathcal{X}_{\text{vert}} &:= \bigcup \{ \{X_1(v, w), X_2(v, w)\} : v, w \in B_{\text{vert}}, r_v \neq r_w \}, \\ \mathcal{X}_{\text{hori}} &:= \bigcup \{ \{X_1(h, k), X_2(h, k)\} : h, k \in B_{\text{hori}}, c_h \neq c_k \}. \end{aligned}$$

3. The case with **two different broken vertices** $h = (r_h, c_h) \in B_{\text{hori}}$ and $v = (r_v, c_v) \in B_{\text{vert}}$ is different to the ones before. As shown in Figure 3.27(a), we have four different cases depending on the relational position of the two vertices, whether the horizontal vertex is above or below,

I) $r_h < u(r_v)$ or

II) $r_h > u(r_v)$,

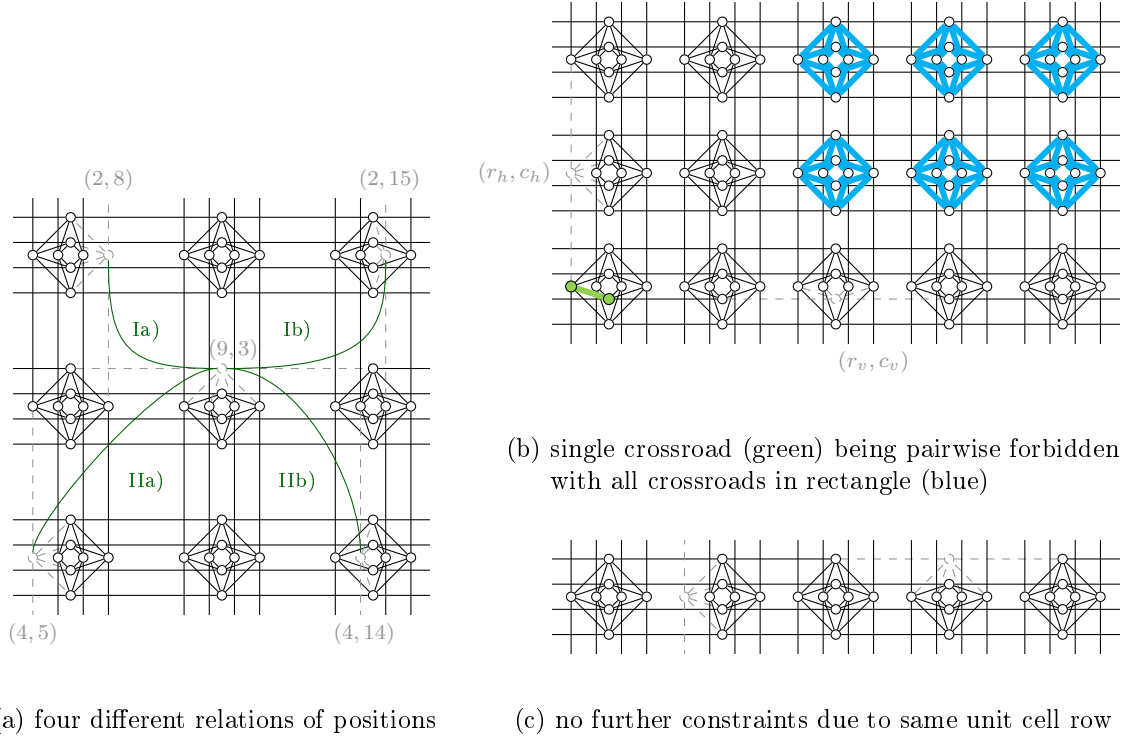


Figure 3.27: Mutually exclusive crossroads caused by two different broken vertices

and left or right,

- a) $c_v > u(c_h)$ or
- b) $c_v < u(c_h)$,

of the vertical broken vertex.

Figure 3.27(b) shows the combination **Ia**) exemplary. The other cases **Ib**), **IIa**) and **IIb**) can be derived analogously but mirrored to different corners. This is covered by the definition of I , which we use again in the following construction; therefore, this holds for all cases. Further, we can see in Figure 3.27(c) that no additional constraints are provided if both vertices lie in the same unit cell row or column, respectively. Therefore, we can restrict on cases with $r_h \neq u(r_v)$ and $c_v \neq u(c_h)$.

Due to the path interruption by the broken vertices, we get exactly one crossroad,

$$r_v c_h \cong (u(r_v), c_h, r_v, u(c_h)),$$

illustrated in green in the lower left corner of Figure 3.27(b), which is pairwise forbidden with all the crossroads in a rectangle, which are shown in blue. In the following, we refer to $r_v c_h$ as the *common crossroad*. In the case shown in Ia), the rectangle includes all rows from the upper boundary until the unit cell of the broken horizontal vertex h and all columns starting at the unit cell of the broken vertical vertex v until the right boundary, which are the combinations in $[4r_h] \times [4(c_v - 1) + 1; n]$. More generally, the rectangle is described by

$$I(r_h, u(r_v)) \times I(c_v, u(c_h)).$$

The pairwise constraints

$$x_{r_v c_h} + x_{rc} \leq 1 \quad \forall rc \in I(r_h, u(r_v)) \times I(c_v, u(c_h))$$

would therefore be sufficient to describe our problem. But taking advantage of the matching constraints (3.1) again, we can aggregate the crossroads in the rectangle: either all in one inner row or all in one inner column. To keep the optimization problem description as small as possible, we take the smallest number of resulting MES. This is given by the minimum of the dimensions of the rectangle, hence the number of rows $|I(r_h, u(r_v))|$ or the number of columns $|I(c_v, u(c_h))|$. With

$$\mathbf{X}_r(\mathbf{h}, \mathbf{v}) := \{r_v c_h\} \cup (\{r\} \times I(c_v, u(c_h))),$$

describing the aggregated MES for a row $r \in I(r_h, u(r_v))$, and analogously

$$\mathbf{X}_c(\mathbf{h}, \mathbf{v}) := \{r_v c_h\} \cup (I(r_h, u(r_v)) \times \{c\}),$$

for a column $c \in I(c_v, u(c_h))$, we can define

$$\mathcal{X}_{\text{mix}}(\mathbf{h}, \mathbf{v}) := \begin{cases} \{X_r(h, v) : r \in I(r_h, u(r_v))\} & \text{if } |I(r_h, u(r_v))| \leq |I(c_v, u(c_h))|, \\ \{X_c(h, v) : c \in I(c_v, u(c_h))\} & \text{otherwise,} \end{cases}$$

choosing the set of MES with the smallest cardinality for a certain pair of broken vertices v and h . With

$$\mathcal{X}_{\text{mix}} := \bigcup \{ \mathcal{X}_{\text{mix}}(h, v) : (h, v) \in B_{\text{hori}} \times B_{\text{vert}}, r_h \neq u(r_v), c_v \neq u(c_h) \},$$

we can finally summarize all cardinality constraints to

$$\sum_{rc \in X} x_{rc} \leq 1 \quad \forall X \in \mathcal{X}_{\text{hori}} \cup \mathcal{X}_{\text{vert}} \cup \mathcal{X}_{\text{mix}}. \quad (3.4)$$

Embedding ILP in a Nutshell

With the definitions of the section before, we can summarize the complete embedding problem in the following ILP formulation. If we find an optimal solution to this ILP, its objective value corresponds to the size of the largest embeddable complete graph using our scheme and the activated variables define the crossroads for a corresponding embedding.

LARGEST COMPLETE GRAPH MATCHING PROBLEM. Given a broken Chimera graph, find x that solves

$$\begin{aligned} \max \quad & \sum_{rc \in N^2} x_{rc} \\ \text{s.t.} \quad & \sum_{r \in N} x_{r\tilde{c}} \leq 1 \quad \forall \tilde{c} \in N, \\ & \sum_{c \in N} x_{\tilde{r}c} \leq 1 \quad \forall \tilde{r} \in N, \\ & \sum_{rc \in X} x_{rc} \leq 1 \quad \forall X \in \mathcal{X}_{\text{hori}} \cup \mathcal{X}_{\text{vert}} \cup \mathcal{X}_{\text{mix}}, \\ & x_{rc} = 0 \quad \forall rc \in N^2 \setminus A, \\ & x \in \{0, 1\}^{N \times N}. \end{aligned}$$

Embedding Extraction

Once a solution $x^* \in \{0, 1\}^{N \times N}$ for the ILP shown in the previous section is found, we can construct the actual embedding from the activated crossroads. For this we extend the crossroad in the corresponding row to the left and the right and in the corresponding column to the top and the bottom until we meet a broken vertex or the boundary of the Chimera to obtain the corresponding cross.

For a crossroad $rc \in N^2$ with $x_{rc}^* = 1$, the vertices to the left of the crossroad are vertical vertices located in the same row r , while the column index ranges from 1 to $u(c) - 1$. More formally, this means the set $\{r\} \times [1; u(c) - 1]$. Remember $u(c)$ defines the unit cell column index corresponding to the inner column index. If there is at least one broken vertex, we need to find the rightmost one among all of them because we cannot extend the cross to the left beyond it. The rightmost broken vertical vertex has the largest column index and can thus be found with

$$\max \{ \tilde{c} \in [1; u(c) - 1] : (r, \tilde{c}) \in B_{\text{vert}} \}.$$

Since the cross does not include the broken vertex, we need to introduce a shift of 1 and therefore can obtain the vertices in the left part of the cross with

$$P_{\text{left}}(\mathbf{r}, \mathbf{c}) := \max \{ \tilde{c} \in [1; u(c)] : \tilde{c} = 1 \vee (r, \tilde{c} - 1) \in B_{\text{vert}} \},$$

where the disjunction with $\tilde{c} = 1$ is needed to default to the start value 1 of the index range in case the set of broken vertices is empty. Analogously, we can construct the parts to the right, top and bottom with

$$\begin{aligned} P_{\text{right}}(\mathbf{r}, \mathbf{c}) &:= \min \{ \tilde{c} \in [u(c); s] : \tilde{c} = s \vee (r, \tilde{c} + 1) \in B_{\text{vert}} \}, \\ P_{\text{top}}(\mathbf{r}, \mathbf{c}) &:= \max \{ \tilde{r} \in [1; u(r)] : \tilde{r} = 1 \vee (\tilde{r} - 1, c) \in B_{\text{hori}} \}, \\ P_{\text{bot}}(\mathbf{r}, \mathbf{c}) &:= \min \{ \tilde{r} \in [u(r); s] : \tilde{r} = s \vee (\tilde{r} + 1, c) \in B_{\text{hori}} \}. \end{aligned}$$

All in all, we obtain the plus-shaped vertex set with

$$(\{r\} \times [P_{\text{left}}(r, c); P_{\text{right}}(r, c)]) \cup ([P_{\text{top}}(r, c); P_{\text{bot}}(r, c)] \times \{c\}),$$

for all $rc \in N^2$ with $x_{rc}^* = 1$.

As an equivalent to the chain length, described in Section 3.3.1, an important parameter in the further processing of the Ising model that we can observe is the *cross size*. The maximum number of vertices in a cross in a Chimera of size s is $2s$, which is only the case when the cross can be extended fully to all boundaries with $P_{\text{left}}(r, c) = P_{\text{top}}(r, c) = 1$ and $P_{\text{right}}(r, c) = P_{\text{bot}}(r, c) = s$.

Clearly, by this construction we obtain embeddings with a larger number of vertices than presented in [7], where the maximum chain length is $s + 1$. However, this allows for larger graph sizes as we see in the following section.

3.3.4 Analysis

In this section, we investigate the structure of the LARGEST COMPLETE GRAPH MATCHING PROBLEM by discussing its size, complexity and variations. The solvability of the problem can be estimated by different parameters. The size of the ILP, more precisely the number of variables and constraints, is of interest when directly passing the constructed ILP to ILP solvers and using them without any further specifications. However, the specific structure of the constraints allows for a deeper complexity analysis of the problem showing fixed-parameter tractability. At the end of this section, we give a short outlook on how the ILP can be extended to more general Chimera graphs.

Size of the ILP

We estimate the size of the ILP with regard to the input parameters s , B_{horiz} and B_{vert} . The number of variables is $n^2 = 16s^2$ if we also take the unavailable combinations in A into account. By removing them, we get $n^2 - |A| \geq n^2 - |B_{\text{horiz}}| - |B_{\text{vert}}|$, where the lower bound is achieved only if no two broken vertices meet in one edge.

Apart from the $2n$ matching constraints in (3.1), we show that the number of additional constraints is also polynomial in the number of broken vertices. We need to count the number of MES that are constructed in the former section for the different combinations of vertices. Taking two unequal vertices out of the broken horizontal vertices B_{horiz} , we get

$$\binom{|B_{\text{horiz}}|}{2} = \frac{1}{2} |B_{\text{horiz}}| (|B_{\text{horiz}}| - 1) = \frac{1}{2} |B_{\text{horiz}}|^2 - \frac{1}{2} |B_{\text{horiz}}|$$

combinations. Analogously, for two broken vertical vertices out of B_{vert} we have

$$\binom{|B_{\text{vert}}|}{2} = \frac{1}{2} |B_{\text{vert}}|^2 - \frac{1}{2} |B_{\text{vert}}|$$

combinations. On the other hand, the number of combinations for two different broken vertices is $|B_{\text{horiz}}| |B_{\text{vert}}|$. Those numbers could be slightly but not significantly reduced when taking pairs into account that lie in the same rows, respectively, columns.

For each combination of two broken vertices of the same type we have two constraints. Thus, we have

$$\begin{aligned} |\mathcal{X}_{\text{horiz}}| &\leq |B_{\text{horiz}}|^2 - |B_{\text{horiz}}|, \\ |\mathcal{X}_{\text{vert}}| &\leq |B_{\text{vert}}|^2 - |B_{\text{vert}}|. \end{aligned}$$

For the different broken vertices, the number of constraints depends on the size of the corresponding rectangles. Here, we can only estimate the worst-case scenario, that is, $n - 1$ constraints, hence

$$|\mathcal{X}_{\text{mix}}| \leq (n - 1) |B_{\text{horiz}}| |B_{\text{vert}}|.$$

Therefore, we get a total of

$$\begin{aligned} &|\mathcal{X}_{\text{horiz}}| + |\mathcal{X}_{\text{vert}}| + |\mathcal{X}_{\text{mix}}| \\ &\leq |B_{\text{horiz}}|^2 - |B_{\text{horiz}}| + |B_{\text{vert}}|^2 - |B_{\text{vert}}| + (n - 1) |B_{\text{horiz}}| |B_{\text{vert}}| \end{aligned}$$

additional cardinality constraints in (3.4).

Problem Complexity

The described problem is a matching problem on a bipartite graph. The simple version, without additional constraints, can be solved in polynomial time, e.g. with the algorithm of Hopcroft and Karp in $\mathcal{O}(n^{2.5})$ [26]. Due to the constraints (3.4), introduced in Section 3.3.3, our ILP corresponds to what is known as a RESTRICTED MAXIMUM MATCHING PROBLEM. Those problems are NP-hard in general, and this even holds for cardinality constraints with a cardinality of just 1 like ours [50]. But by exploiting the specific structure of those constraints, we can derive that the runtime is mainly dominated by the broken vertices compared to the size of the Chimera graph. More formally, this means:

Theorem 3.38. *The LARGEST COMPLETE GRAPH MATCHING PROBLEM is fixed-parameter tractable in the number of the broken vertices $|B|$.*

Proof. We show the fixed-parameter tractability by enumerating the decisions that have to be made for removing constraints until the problem is a simple MAXIMUM BIPARTITE MATCHING PROBLEM.

Considering the constraint for two broken vertices of the same type, we have two MES in (3.2), respectively, (3.3). As it is shown in Section 3.3.3, both MES consist of a left and a right part lying in different rows for broken vertical vertices, respectively, an upper and a lower part in different columns for broken horizontal vertices. Since we can only take one of the crossroads in an MES into a solution, this crossroad is either in the left or in the right, respectively, upper or lower, part. Imagine we decide in advance for one part of the MES. Considering, for instance, some $X \in \mathcal{X}_{\text{vert}}$, with $X =: X_{\text{left}} \cup X_{\text{right}}$ for simplicity, we could choose the crossroad to be in X_{left} . Thus, none of the crossroads in X_{right} can be activated in the solution, which means we have to set $x_{rc} = 0$ for all $rc \in X_{\text{right}}$. The corresponding cardinality constraint reduces to

$$\sum_{rc \in X} x_{rc} = \sum_{rc \in X_{\text{left}}} x_{rc} \leq 1.$$

As X_{left} only consists of crossroads in a certain row, the above constraint is weaker than the matching constraint of (3.1) covering this row fully. Thus, we can remove it and the resulting optimization problem, having less variables and less constraints, is easier to solve.

By considering both exclusive options, disregarding X_{right} or X_{left} , and choosing the best solution, we get the global optimum. This procedure can be applied for every MES in $\mathcal{X}_{\text{hori}}$ and $\mathcal{X}_{\text{vert}}$, especially this can be done iteratively to already simplified versions. With two parts for each MES, this results in total in

$$2^{|\mathcal{X}_{\text{hori}}|} \cdot 2^{|\mathcal{X}_{\text{vert}}|} \leq 2^{|B_{\text{hori}}|^2 - |B_{\text{hori}}| + |B_{\text{vert}}|^2 - |B_{\text{vert}}|}$$

different simplified problems.

For different broken vertices, we have much more constraints, but they all have one crossroad in common that cannot be matched together with the other concerned crossroads in the rectangle. Therefore, we can proceed similarly to before. One option is just taking the single common crossroad into the solution and rejecting all of the rectangle. This means that the binary variable corresponding to this crossroad is set to 1, while those for the rectangle are set to 0. By this not only the constraints are removed, but the size of the ILP is appreciably reduced, persisting for all problems resulting from subsequent decisions. However, for an increasing size of the rectangle it gets more unlikely that the common crossroad is part of an optimal solution. Hence, the second option is rejecting this single crossroad. This again results in weaker remaining constraints than the matching constraints for the whole rectangle, and they can be dropped. These two possibilities per broken vertex pair result in total in further $2^{|B_{\text{hori}}| |B_{\text{vert}}|}$ options.

Finally, we get at maximum 2^p , with

$$p = |B_{\text{hori}}| |B_{\text{vert}}| + |B_{\text{hori}}|^2 - |B_{\text{hori}}| + |B_{\text{vert}}|^2 - |B_{\text{vert}}|,$$

different simplified versions of our original problem. As we removed all of the additional constraints along the decision tree, they are now simple MAXIMUM BIPARTITE MATCHING PROBLEMS

and can be solved efficiently. With

$$\begin{aligned} p &\leq |B_{\text{hori}}|^2 + |B_{\text{vert}}|^2 + 2|B_{\text{hori}}||B_{\text{vert}}| \\ &= (|B_{\text{hori}}| + |B_{\text{vert}}|)^2 \\ &= |B|^2 \end{aligned}$$

we get a worst-case runtime in

$$\mathcal{O}\left(2^{|B|^2} n^{2.5}\right)$$

and the problem is fixed-parameter tractable in the choice of the broken vertices. \square

According to current hardware development, we can reasonably assume that $|B|$ is small compared to n . Therefore, considering $|B|$ to be fixed, the problem is efficiently solvable for increasing size n . However, this means at the same time that the ratio of broken vertices R is decreasing because it is inversely proportional to n^2 . Thus, considering the ongoing development of the annealing machines, the exponential dependency of the runtime on the number of broken qubits will be negligible in contrast to the just polynomial dependency on the size of the Chimera.

Just keeping R fixed, like we did in our experiments for comparison with [7], still provides an exponential runtime. This aspect demands for heuristic solving approaches, like we present in the following section. However, once the embedding is computed for a hardware graph, it can be reused during the whole operating period. This justifies a larger runtime than in the operational approach calculating a new embedding for each Ising instance.

Generalization

In Section 3.3.3, we show the construction of the ILP for the LARGEST COMPLETE GRAPH MATCHING PROBLEM exemplarily for the symmetric Chimera graph with a depth of 4. But the whole setup can also be generalized for arbitrary Chimera graphs \mathcal{C}_{rad} , which thus are rectangular, meaning $r \neq c$, or have a different depth d . In the following, we briefly mention the adjustments that need to be applied.

If d is different than 4, the unit cell index function changes to $u: x \mapsto \lceil \frac{x}{d} \rceil$. In case of $r \neq c$, we need to split N^2 up in $R \times C$ with the row, respectively, column sets $R = [dr]$ and $C = [dc]$. Of course, the maximal number of vertices in an embeddable complete graph, even if it is an ideal Chimera, is just $d \min\{r, c\}$ in this case. As the amount of rows and columns is not equal anymore, we have to adjust the interval function to be able to differ between maximal row or column with

$$I_X(s_1, s_2) := \begin{cases} [4s_1] & \text{if } s_1 \leq s_2, \\ [4s_1 - 3; |X|] & \text{otherwise} \end{cases}$$

for $X \in \{R, C\}$. With these modifications, it is possible to construct the analogous matching constraints as well as the MES for the cardinality constraints. Similarly, the extraction of the embedding from a found solution can be adjusted.

One might also consider to extend the model by adding broken edges, which could possibly be handled in a similar case differentiation as for the broken vertices. But in contrast to the restriction to broken vertices the implications on the model construction and therefore the size and complexity are not trivial. Further, a broken edge adjoining non-broken vertices is very rare and can thus be handled by marking one of the concerned vertices as broken. Therefore, we do not discuss this in more detail here.

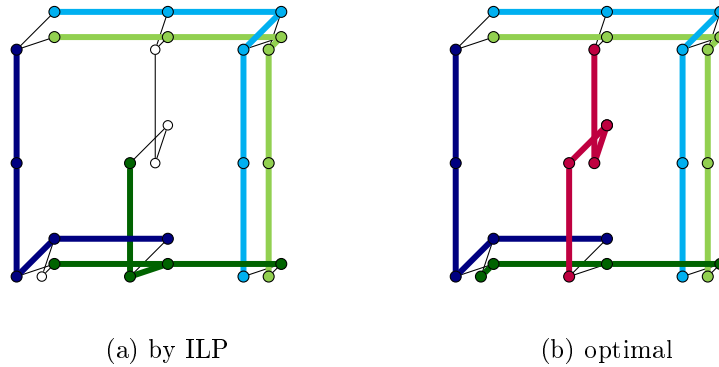


Figure 3.28: Solutions for a very broken Chimera graph containing a larger complete graph than can be found with our ILP

Delineation

In our construction, the embedding corresponding to a single logical vertex is formed by a cross. In the case of many broken vertices, this assumption might be too restrictive. An example is shown in Figure 3.28, where the solution to the ILP formulation from the LARGEST COMPLETE GRAPH MATCHING PROBLEM is not as good as the optimal solution to the LARGEST COMPLETE GRAPH EMBEDDING PROBLEM. We however believe that such corner cases are of less practical relevance since they just seem to occur for a very large ratio of broken vertices.

3.3.5 Heuristic ILP

The complexity analysis in Section 3.3.4 has shown a certain structure of the additional constraints. Especially for the case of two broken vertices of different types there is a strong imbalance: Activating the single common crossroad excludes all the crossroad in the corresponding rectangle. Thus, it is very unlikely that this crossroad is part of the optimal solution, in particular, for growing size of the rectangle. In this section, we show the derivation of a simpler ILP whose solution is assumed to be close to the optimal one.

Reducing Size

We decided to test a heuristic approach based on excluding such unlikely common crossroads in advance. This reduces the number of variables and more importantly the number of constraints. Thus, we solve only a certain part of the decision tree constructed in Section 3.3.4 and therefore, it is not clear whether the optimal value can be achieved.

We introduce a parameter defining which common crossroads shall be removed, respectively, kept: The *maximum rectangle ratio*, denoted here by m with $0 \leq m \leq 1$, gives a boundary on the size of the rectangle relative to the Chimera graph size s below which the common crossroad is kept. If the number of unit cell rows times the number of columns of the rectangle exceeds ms^2 , the crossroad is excluded. More formally, this means, for two different broken vertices $h = (r_h, c_h) \in B_{\text{hori}}$ and $v = (r_v, c_v) \in B_{\text{vert}}$, we do not use the crossroad $r_v c_h$, hence set $x_{r_v c_h} = 0$ in advance, if we have $M(h, v) \geq ms^2$ with

$$M(\mathbf{h}, \mathbf{v}) := |I(r_h, u(r_v))| \cdot |I(c_v, u(c_h))|.$$

Thus, a ratio of 1 means that all common crossroads are kept, while a ratio of 0 means that none of them remain in the resulting optimization problem.

Given m , let the set of unused crossroads be

$$U^m := \{r_v c_h : (h, v) \in B_{\text{hori}} \times B_{\text{vert}}, M(h, v) \geq ms^2\}$$

and further let

$$\mathcal{X}_{\text{mix}}^m := \bigcup \{\mathcal{X}_{\text{mix}}(h, v) : (h, v) \in B_{\text{hori}} \times B_{\text{vert}}, r_h \neq u(r_v), c_v \neq u(c_h), M(h, v) < ms^2\}.$$

be the reduced set of MES. With this the corresponding constraints can be simplified by replacing \mathcal{X}_{mix} with $\mathcal{X}_{\text{mix}}^m$ in the LARGEST COMPLETE GRAPH MATCHING PROBLEM. This can be seen in the following section summarizing the heuristic ILP.

Heuristic Embedding ILP in a nutshell

With the same definitions as before and a certain choice of m , we can now summarize the heuristically reduced embedding problem in the ILP formulation:

HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM. Given a broken Chimera graph and ratio $m \in \mathbb{R}$ with $0 \leq m \leq 1$, find x that solves

$$\begin{aligned} \max \quad & \sum_{rc \in N^2} x_{rc} \\ \text{s.t.} \quad & \sum_{r \in N} x_{r\tilde{c}} \leq 1 \quad \forall \tilde{c} \in N, \\ & \sum_{c \in N} x_{\tilde{r}c} \leq 1 \quad \forall \tilde{r} \in N, \\ & \sum_{rc \in X} x_{rc} \leq 1 \quad \forall X \in \mathcal{X}_{\text{hori}} \cup \mathcal{X}_{\text{vert}} \cup \mathcal{X}_{\text{mix}}^m, \\ & x_{rc} = 0 \quad \forall rc \in N^2 \setminus A \cup U^m, \\ & x \in \{0, 1\}^{N \times N}. \end{aligned}$$

3.3.6 Experimental Setup

Random Instances

To be able to compare our approach to current state-of-the-art methods, we consider different ratios of broken vertices for growing hardware sizes. We have generated ten instances for each combination of the following values:

- sizes of Chimera graph: $s \in \{4, 6, 8, \dots, 32, 34\}$,
- ratios of broken vertices: $b \in \{0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.1, 0.2\}$.

The ratio of the broken vertices times the total number of vertices in the ideal Chimera graph results in the number of broken vertices for a certain size. For each of the ten instances, we randomly chose this number out of all vertices and marked them as being broken. Due to rounding to whole vertices, the resulting exact ratios differ slightly from the aimed ones, especially for smaller graph sizes.

As a reference, we like to remark the parameters of two real D-Wave 2000Q systems. First, the solver DW_2000Q_6, which we accessed through the Jülich UNified Infrastructure for Quantum computing (JUNIQU), has a size of 16 and 7 broken vertices. This corresponds to a ratio of about 0.0034. Second, the older USRA/NASA chip with the same size had 17 broken vertices, resulting in a ratio of about 0.0083 [30]. Thus, our experiments are much more exhaustive than current hardware demands.

For the heuristic approach, we use $m = 0$ and $m = 0.25$ for our experiments as reference points to evaluate the impact of removing a significant number of crossroads.

Solving Strategy

This work focuses on the presentation of the embedding problem as an ILP. We did not implement an algorithm, yet, which exploits the branching procedure as described in Section 3.3.4. It is not straightforward how the decision tree could be reduced at certain stages. As we do not consider the ratio of broken vertices to be fixed here, there is still an exponential overhead in the number of final simplified problems. Even the simplified heuristic version is still a hard optimization problem.

Thus, using an ILP solver, already taking advantage of implemented branch-and-bound techniques, is a good starting point to evaluate the capabilities of the model. We decided to pass the models of the LARGEST COMPLETE GRAPH MATCHING PROBLEM and the HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM directly to the solver SCIP [21] without further adjustments. It is currently one of the fastest non-commercial solvers for mixed integer programming, which includes ILP.

Specifications

The experiments were run on a Dell Precision 5820 Tower workstation with a Intel Xeon(R) W-2175 CPU @ 2.50GHz \times 28, 128 RAM and operating system Ubuntu Linux 18.04.5 LTS. We implemented our code in python and used the python interface package `pyscipopt` [39] to connect to the solver SCIP with version 6.0.1 [21]. As this interface does not support parallel mode, we could just use one core. We set a timeout of 1 hour for solving each instance with SCIP. Building up the model was not included in there. As a start, we evaluate the capabilities of the model and the derived heuristic version themselves. Thus, we did not optimize our code regarding performance. Apart from the timeout, we used the default SCIP parameters.

3.3.7 Results

A good reference to estimate the quality of a solution is to compare its objective value, the found graph size, to the largest possible size of a complete graph in the ideal Chimera graph. As stated in Section 3.3.1 with our construction using crosses, the largest complete graph in a Chimera graph of size s is K_{4s} . Let $G_{s,b,i}$ be the graph size returned by SCIP within one hour for the

i th instance with Chimera size s and ratio of broken vertices b . As we consider ten instances for each parameter combination, the found graph sizes are averaged and we introduce the *averaged solution ratio*

$$\bar{R}_{s,b} := \frac{\frac{1}{10} \sum_{i=0}^9 G_{s,b,i}}{4s} = \sum_{i=0}^9 \frac{G_{s,b,i}}{40s}$$

as a measure for the solution quality. Table 3.1 shows the resulting ratios for each of the models.

In Table 3.1(a), we can see a clear boundary between the instances which can be solved in one hour and which cannot. The former are either instances with a small Chimera size or with a small ratio of broken vertices. Here, we already see a slight advantage in the achieved graph sizes over [7]. Besides, we also solved the exact model for both versions of the aforementioned D-Wave 2000Q chips with 7 and 17 broken vertices, respectively. Not surprisingly in accordance with the results of Table 3.1(a), we were able to find an embedding of the complete graph with 64 vertices in both cases.

For the unsolved instances, we use the current best solution SCIP provides at the timeout, being a proven lower bound on the actual optimum. As SCIP is a MIP solver, it tries to solve a given model to proven optimality and is thus not made for calculating fast approximate solutions. Therefore, the found solution values for instances with increasing Chimera sizes and ratios decrease significantly, due to the sizes of the models. The instance combinations (32, 0.2) and (34, 0.2) could not be solved at all with the exact model because SCIP ran out of memory. Nevertheless, the remaining instances provide values comparable to those of [7].

Evaluating the heuristic approach, we can see that Table 3.1(b) for $m = 0.25$ does not differ significantly from the one for the exact model according to solvability. Having a closer look at the values, we see no decrease for the solved instances. However, there is a slight improvement for the unsolved instances. Thus, the model has a slight advantage regarding runtime but still seems to yield close to optimal values.

Regarding the heuristic approach with $m = 0$, shown in Table 3.1(c), we have a much stronger difference to the exact model. A lot more instances could be solved within one hour of computation time, especially those with larger ratios of broken vertices. For the combination of sizes above 18 and ratios between about 0.02 and 0.05, we see a clear improvement through the heuristic, although a few instances could not be solved, too. This region of parameters, where it is reasonable to use this heuristic, is also clearly recognizable in Table 3.2, where we show the advantage of the heuristics with either $m = 0$ or $m = 0.25$.

However, in Table 3.1(c), we observe that the proportions of graph sizes are much smaller for ratios above 0.1 than for $m = 0.25$. Thus, this heuristic model seems to get easier again with an increasing ratio of broken vertices. We assume a significant number of crossroads is excluded in advance, because of the large number of broken vertices, such that the resulting model has only very few solutions left. In these cases, the heuristic with $m = 0$ is much too restrictive and $m = 0.25$ is advantageous.

In order to compare our approach to previous work by Boothby et al. [7], we similarly plot the found graph sizes for selected ratios of broken vertices in Figure 3.29 on page 62. For larger ratios of broken vertices, e.g. $b = 0.1$ and $b = 0.05$, the maximum over the found graph sizes is comparable to [7] for both $m = 0.0$ and $m = 0.25$. In contrast, for smaller ratios of broken vertices, e.g. $b = 0.01$ and $b = 0.02$, our heuristic approach with $m = 0.0$ is able to embed larger complete graphs than it was reported in [7]. Note that the diagonal corresponds to representing the largest possible complete graph size.

$b \backslash s$	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
0.005	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.01	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.97	0.91	0.82	0.85
0.02	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.95	0.77	0.68	0.70	0.63	0.60	0.60
0.03	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.63	0.59	0.59	0.58	0.51	0.51	0.52	0.44
0.04	1.00	1.00	1.00	0.99	0.99	0.99	0.88	0.60	0.61	0.56	0.50	0.52	0.43	0.44	0.43	0.39
0.05	0.99	0.99	0.98	0.99	0.98	0.94	0.65	0.59	0.43	0.51	0.44	0.43	0.38	0.37	0.35	0.38
0.1	0.95	0.93	0.86	0.81	0.69	0.52	0.37	0.35	0.30	0.29	0.25	0.25	0.24	0.22	0.21	0.17
0.2	0.72	0.60	0.53	0.48	0.39	0.22	0.20	0.16	0.17	0.17	0.12	0.14	0.11	0.10	-	-

(a) for the LARGEST COMPLETE GRAPH MATCHING PROBLEM

$b \backslash s$	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
0.005	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.01	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.88	0.87	0.86
0.02	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.75	0.76	0.77	0.73	0.69
0.03	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.78	0.70	0.67	0.67	0.62	0.62	0.61	0.59
0.04	1.00	1.00	1.00	0.99	0.99	0.99	0.97	0.74	0.70	0.63	0.61	0.60	0.56	0.53	0.49	0.48
0.05	0.99	0.99	0.98	0.99	0.98	0.95	0.72	0.67	0.60	0.57	0.52	0.50	0.49	0.45	0.42	0.41
0.1	0.95	0.93	0.86	0.81	0.71	0.59	0.43	0.41	0.36	0.32	0.31	0.29	0.28	0.23	0.24	0.21
0.2	0.72	0.60	0.52	0.48	0.42	0.36	0.31	0.22	0.17	0.16	0.13	0.12	0.11	0.09	0.09	0.08

(b) for the HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM with $m = 0.25$

$b \backslash s$	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
0.005	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.01	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.02	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.98	0.97	0.92
0.03	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.96	0.94	0.89	0.86	0.81	0.79	0.74	0.69
0.04	1.00	1.00	1.00	0.99	0.99	0.98	0.96	0.90	0.84	0.79	0.75	0.68	0.62	0.58	0.53	0.49
0.05	0.99	0.99	0.98	0.99	0.96	0.91	0.82	0.78	0.68	0.63	0.57	0.50	0.45	0.39	0.37	0.35
0.1	0.95	0.90	0.77	0.67	0.51	0.43	0.37	0.29	0.26	0.19	0.15	0.14	0.11	0.08	0.06	0.06
0.2	0.66	0.43	0.28	0.22	0.13	0.09	0.06	0.03	0.02	0.01	0.01	0.01	0.00	0.01	0.00	0.00

(c) for the HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM with $m = 0.0$

Table 3.1: Averaged solution ratios $\bar{R}_{s,b}$ for each combination of size s and ratio of broken vertices b with number of instances that where solved to optimality (white: all 10, yellow: 1 to 9, red: none).

$b \backslash s$	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
0.005	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.01	-0.12	-0.13	-0.14
0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.06	-0.17	-0.25	-0.23	-0.21	-0.24	-0.23
0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.18	-0.24	-0.22	-0.19	-0.19	-0.17	-0.08	-0.10
0.04	0.00	0.00	0.00	0.00	0.00	0.01	0.01	-0.16	-0.14	-0.16	-0.14	-0.08	-0.06	-0.05	-0.04	-0.01
0.05	0.00	0.00	0.00	0.00	0.02	0.04	-0.10	-0.11	-0.08	-0.06	-0.05	0.00	0.04	0.06	0.05	0.06
0.1	0.00	0.03	0.09	0.14	0.20	0.16	0.06	0.12	0.10	0.13	0.16	0.15	0.17	0.15	0.18	0.15
0.2	0.06	0.17	0.24	0.25	0.29	0.27	0.25	0.19	0.15	0.15	0.12	0.11	0.11	0.08	0.09	0.08

Table 3.2: Difference of rounded averaged solution ratios from heuristic models showing the advantage of a certain model (green: $m = 0.25$, blue: $m = 0.0$).

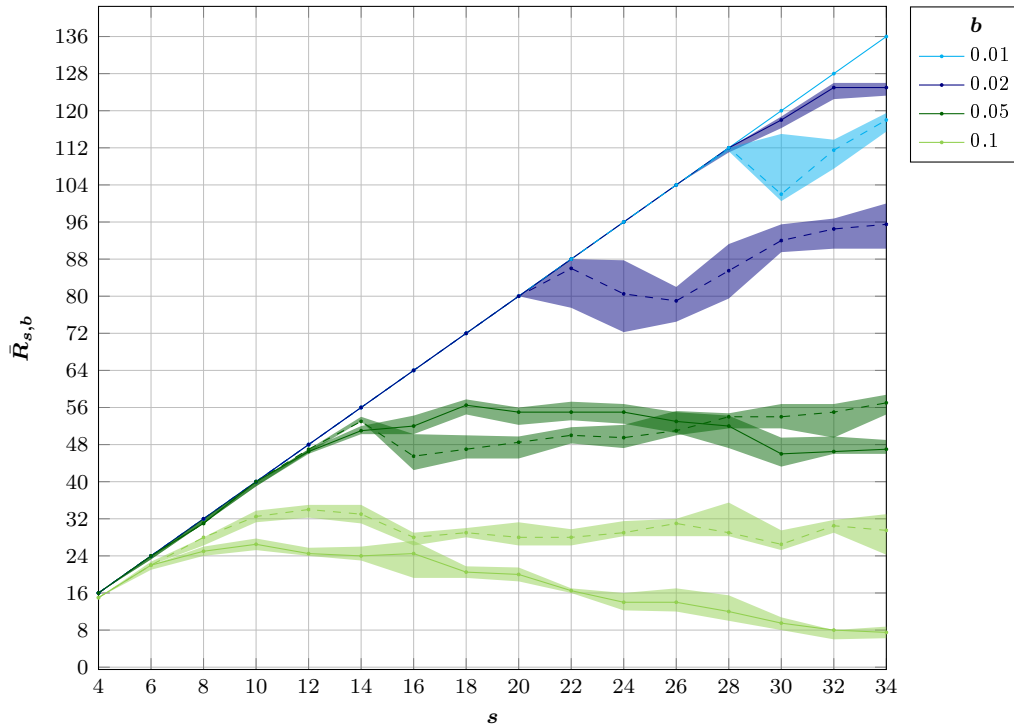


Figure 3.29: Complete graph sizes against Chimera sizes s for selected ratios of broken vertices b for the HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM) illustrated by the median of $G_{s,b,i}$ over $i \in [10]$ for $m = 0.0$ (solid lines) and $m = 0.25$ (dashed lines) with quartiles (shaped regions)

For a ratio of 0.05 or smaller, we observe that the proportions from the solved instances with small size and ratio have a value very close to 1.0, which means that most of them yield a maximal or close to maximal complete graph despite the presence of broken vertices. Due to the heuristic results, we expect just a very small decline in the proportions for the exact model for larger Chimera sizes, if we could solve them to the end. This is based on the fact that the heuristics provide a lower bound on the actual optimum of the exact model. Thus, despite the shortcomings presented in Section 3.3.4, our model is indeed very powerful.

All in all, we see that the formulated LARGEST COMPLETE GRAPH MATCHING PROBLEM can be solved to optimality using state-of-the-art MIP solvers for small Chimera sizes or a small ratio of broken vertices. Especially in these parameter settings, the optimal value of the heuristic version, the HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM, does not differ significantly from the original one. For larger Chimera graphs with a ratio of broken vertices in a certain range, the heuristic performs even better within the given time constraint of one hour, due to the removal of unlikely crossroads and thus several constraints. However, if the ratio is too large, here above 0.1, the heuristic is too restrictive and the solution quality decreases again. Nevertheless, the complete graph sizes we have found exceed the ones from previous approaches [32, 7].

4 Weight Distribution

With only the embedding, we still cannot run experiments on the D-Wave machine. We need to find suitable parameters for an Ising model working on the hardware graph that represents the original Ising model which we actually want to solve. As this requires to split the weight of an original vertex over several hardware vertices, we use the term ‘weight distribution’ to describe the problem, which is handled in this chapter.

We start in Section 4.1 by setting up the full embedded Ising problem and breaking it down into smaller problems: By extracting the part concerning a single vertex, we derive sufficient requirements on the parameters concerning this vertex. We formulate and simplify the corresponding resulting optimization problems with different objectives in Section 4.2. After the investigation of these problems for two special cases in Section 4.3, for instance, in case we have a total weight of 0, we establish a simplified description of the polyhedron over which the optimization problems are defined in Section 4.4. In Sections 4.5 and 4.6, we deeply analyze the solvability of what we call the ‘strength-only’ and the ‘full’ variant of the optimization problem, respectively. Finally, we transfer the results which we have found for the continuous problems into the integer programming context in Section 4.7.

4.1 Problem Extraction

After the pure graph embedding, handled in the previous chapter, is found, the parameters of the Ising model need to be transferred as well. For this we introduce the corresponding *embedded Ising model* and the related concepts necessary when dealing with the D-Wave annealing machine. The main basis for this was formed by V. Choi in [12] and [13].

We show that, by synchronizing all vertices in the hardware corresponding to a vertex of the original problem interaction graph, we can construct an Ising model that represents the original one in the specific hardware structure. By observing a single vertex, we can extract a specific optimization problem, which is investigated in detail in the next sections.

4.1.1 Embedded Ising Model

The Ising model as given in Definition 2.1 is defined over arbitrary graphs, thus also over the possible hardware graphs. As explained before, typical applications however need an embedding. Therefore, we introduce an extended definition of the Ising model in this section to combine both concepts. For this we first extend the embedding notation, remember Definition 3.3, by the following graph structures:

Definition 4.1. For two graphs G and H and the embedding $\varphi : V(G) \rightarrow 2^{V(H)}$, let the *embedded graph*, the subgraph of H resulting from the embedding, be

$$\mathbf{H}_\varphi := H \left[\bigcup_{v \in V(G)} \varphi_v \right] = \left(\bigcup_{v \in V(G)} \varphi_v, E_\varphi \cup E_\delta \right)$$

with

$$\begin{aligned} \mathbf{E}_\varphi &:= \bigcup_{v \in V(G)} E(H[\varphi_v]), \\ \mathbf{E}_\delta &:= \bigcup_{vw \in E(G)} \delta_{vw} := \bigcup_{vw \in E(G)} \delta(\varphi_v, \varphi_w), \end{aligned}$$

denoting the *intra-connecting* and the *inter-connecting edges*, respectively.

Using the embedding objects of Definition 4.1, we can now formulate an Ising model in the given embedded graph. The following concepts are mainly well known in the quantum annealing community, see e.g. [12] and [44], but we want to bring them into a more formal format here.

Definition 4.2. An *embedded Ising model* for two graphs G and H and an embedding φ of G in H is an Ising model over H_φ , where we have $\bar{\mathbf{I}} : \{-1, 1\}^{V(H_\varphi)} \rightarrow \mathbb{R}$ with

$$\begin{aligned} \bar{\mathbf{I}}(s) &:= \sum_{v \in V(G)} \left(\sum_{q \in \varphi_v} \bar{W}_q s_q + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} s_p s_q \right) + \sum_{vw \in E(G)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} s_p s_q \\ &= \sum_{q \in V(H_\varphi)} \bar{W}_q s_q + \sum_{pq \in E_\varphi \cup E_\delta} \bar{S}_{pq} s_p s_q \end{aligned}$$

for the weights $\bar{\mathbf{W}} \in \mathbb{R}^{V(H_\varphi)}$ and the strengths $\bar{\mathbf{S}} \in \mathbb{R}^{E(H_\varphi)}$.

In this case, we call the corresponding ISING PROBLEM of finding an s that solves

$$\min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{\mathbf{I}}(s)$$

with the above embedded Ising model $\bar{\mathbf{I}}$ the EMBEDDED ISING PROBLEM.

With this formulation and $H = C$ for C being a currently operating broken Chimera graph of D-Wave, we could solve the corresponding EMBEDDED ISING PROBLEM with the D-Wave annealer (with some probability). However, given an arbitrary Ising model whose underlying connectivity graph requires an embedding, we need to find a suitable corresponding embedded Ising model. This requires to choose the weights and strengths in a certain way such that an optimal solution of the new ISING PROBLEM corresponds to an optimal solution of the original one in the end.

In particular, as we usually do not only want to know the optimal value but also the optimal solution itself, we need a recipe how to get from an embedded to an original solution. We therefore need a ‘de-embedding’ function that can be computed easily, which means in polynomial time. This is more formally stated by

Definition 4.3. An *equivalent embedded Ising model* $\bar{I} : \{-1, 1\}^{V(H_\varphi)} \rightarrow \mathbb{R}$ to a given Ising model $I : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ for two graphs G and H and an embedding φ of G in H fulfils the following properties: The corresponding Ising problems are equivalent in the sense that we have

$$\min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{I}(s) + c = \min_{t \in \{-1, 1\}^{V(G)}} I(t)$$

for some constant $c \in \mathbb{R}$ and there exists a mapping from an optimal solution $s^* \in \{-1, 1\}^{V(H_\varphi)}$ of the EMBEDDED ISING PROBLEM to an optimal solution $t^* \in \{-1, 1\}^{V(G)}$ of the (unembedded) ISING PROBLEM which can be computed in polynomial time.

This would have been sufficient to use the quantum annealing machines in practice if the underlying physical system had ideally realized the corresponding physical model. However, this is impossible in the real world and the machines thus only work heuristically providing solutions only with some unknown probability. In general, it remains unclear whether we have found the optimal solution, a sub-optimal solution or no solution at all. Thus, the user does not only need to have access to the mentioned mapping of the optimal solutions but rather needs more information to deal with the results of the machine.

In practice, we need an extended version of the above definition to overcome this issue: For each solution provided by the annealer, not only optimal ones, we want to know whether we can de-embed it to an original solution and if we can, we also want to know how to do it. We define:

Definition 4.4. An *equivalent embedded Ising model* $\bar{I} : \{-1, 1\}^{V(H_\varphi)} \rightarrow \mathbb{R}$ to a given Ising model $I : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ for two graphs G and H and an embedding φ of G in H is called *de-embeddable* if we have two functions

$$\psi : \{-1, 1\}^{V(H_\varphi)} \rightarrow \{0, 1\}$$

and

$$\tau : \{s \in \{-1, 1\}^{V(H_\varphi)} : \psi(s) = 1\} \rightarrow \{-1, 1\}^{V(G)}$$

which can both be computed in polynomial time. While

$$\psi(s) = \begin{cases} 1 & \text{if } s \text{ is de-embeddable,} \\ 0 & \text{otherwise} \end{cases}$$

tells whether we can compute an original solution to the embedded one, the function τ provides the corresponding de-embedded solution, where we have

$$\bar{I}(s) + c = I(\tau(s)) \quad \forall s \in \{-1, 1\}^{V(H_\varphi)} \text{ with } \psi(s) = 1$$

for the constant $c \in \mathbb{R}$.

We call

$$\psi^{-1}(1) = \{s \in \{-1, 1\}^{V(H_\varphi)} : \psi(s) = 1\}$$

the set of *de-embeddable solutions*.

Thus, for

$$s^* \in \arg \min_{s \in \{-1,1\}^{V(H_\varphi)}} \bar{I}(s),$$

we have by Definition 4.3

$$\tau(s^*) \in \arg \min_{t \in \{-1,1\}^{V(G)}} I(t).$$

The most useful in practice would be if all original solutions had a corresponding embedded counterpart, which means if τ is surjective. This in turn would mean we have $\psi^{-1}(1) \cong \{-1, 1\}^{V(G)}$ and at least $2^{|V(G)|}$ solutions that are de-embeddable.

To find such functions, we need to decide at some point what structure the embedded solutions should follow. Although different options might be possible due to the large number of adjustable parameters, the most straightforward way is to restrict the considerations to solutions where all variables corresponding to the embedding of a single original vertex hold the same value. This principle called *synchronization* is explained in the following section in more detail.

4.1.2 Synchronization

The main aspect of the equivalence of the given and the EMBEDDED ISING PROBLEM is the retrieval of the original solution from the embedded one. For this we need to be able to ‘de-embed’ the embedded solution. This in turn requires this solution to hold a certain structure. By enforcing the *synchronization* of all variables in the embedded Ising model that correspond to a single original variable, which means that all those variables should hold the same value, we have a simple criterion on the solutions of the EMBEDDED ISING PROBLEM. This idea was already introduced in [12] and means more formally

Definition 4.5. A solution of the EMBEDDED ISING PROBLEM $s \in \{-1, 1\}^{V(H_\varphi)}$ is called a *synchronized solution* with respect to an embedding φ of G in H for two graphs G and H if we have

$$s_q = t_v \in \{-1, 1\} \quad \forall q \in \varphi_v \quad \forall v \in V(G).$$

For such a synchronized solution, we can easily provide the functions required for the de-embedding with

$$\psi(s) = \begin{cases} 1 & \text{if } s_q = s_p \quad \forall p, q \in \varphi_v \quad \forall v \in V(G), \\ 0 & \text{otherwise} \end{cases}$$

and

$$\tau(s) = s_X$$

for some vertex set $X \subseteq V(H_\varphi)$ with $|X \cap \varphi_v| = 1$ for all $v \in V(G)$. The vertex set X just serves as a placeholder, as we can simply choose a random vertex from φ_v to obtain the value of its variable because all of them hold the same value. It is easy to recognize that τ is surjective and both functions can be computed in polynomial time.

In case the embedded variables do not hold a common value, it is unclear which value to assign to the corresponding original variable. Heuristics such as majority voting might be useful, when considering the non-optimal solutions provided by the D-Wave machine due to its physical ‘imperfectness’. However, if the embedded Ising model is ill-defined, which means that its optimal solution does not yield a clear correspondence to an original solution, those heuristics will not be able to extract the optimal original solution.

Thus, how do we ensure that such an *embedded Ising model based on synchronization*, which means it yields the given functions as a de-embedding, is an equivalent embedded Ising model to our given one? Obviously, the weights and the strengths of the embedded Ising model depend on the original parameters.

If the weights and the strengths fulfil

$$W_v = \sum_{q \in \varphi_v} \bar{W}_q$$

and

$$S_{vw} = \sum_{pq \in \delta_{vw}} \bar{S}_{pq},$$

respectively, we have for a synchronized solution s as given in Definition 4.5

$$\begin{aligned} \bar{I}(s) &= \sum_{v \in V(G)} \left(\sum_{q \in \varphi_v} \bar{W}_q s_q + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} s_p s_q \right) + \sum_{vw \in E(G)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} s_p s_q \\ &= \sum_{v \in V(G)} \left(\sum_{q \in \varphi_v} \bar{W}_q t_v + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} t_v t_v \right) + \sum_{vw \in E(G)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} t_v t_w \\ &= \sum_{v \in V(G)} t_v \left(\sum_{q \in \varphi_v} \bar{W}_q \right) + \sum_{v \in V(G)} \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} + \sum_{vw \in E(G)} t_v t_w \left(\sum_{pq \in \delta_{vw}} \bar{S}_{pq} \right) \\ &= \sum_{v \in V(G)} W_v t_v + \sum_{vw \in E(G)} S_{vw} t_v t_w + \sum_{pq \in E_\delta} \bar{S}_{pq} \\ &= I(t) + \sum_{pq \in E_\delta} \bar{S}_{pq}. \end{aligned} \tag{4.1}$$

This means, for all such synchronized solutions, we have $\bar{I}(s) + c = I(t)$ with

$$c = - \sum_{pq \in E_\varphi} \bar{S}_{pq}.$$

Thus, the strengths \bar{S}_{E_φ} only introduce an offset to the overall objective value for these solutions. Furthermore, we ensure that for an optimal solution

$$t^* \in \arg \min_{t \in \{-1, 1\}^{V(G)}} I(t)$$

we have $\bar{I}(s^*) + c = I(t^*)$ for $s^* = (t_v^* \mathbb{1}_{\varphi_v})_{v \in V(G)}$ and s^* thus also is the minimum over all synchronized solutions, which means

$$s^* \in \arg \min \left\{ \bar{I}(s) : s \in \{-1, 1\}^{V(H_\varphi)}, s_{\varphi_v} \in \{-1, 1\} \forall v \in V \right\}.$$

However, for the given s^* , we do not necessarily have

$$s^* \in \arg \min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{I}(s),$$

which means it would also be the optimum over all solutions of the EMBEDDED ISING PROBLEM. There might be unsynchronized variable assignments yielding a lower objective value. This is the case if the contribution of the inter-connecting edges does not suffice.

As it can be seen in (4.1), if the variables s_q and s_p for $pq \in E(H[\varphi_v])$ are synchronized, their product reduces to 1 and the corresponding strength \bar{S}_{pq} is added to the objective value. In turn, if the variables are assigned to different values, the product is -1 and \bar{S}_{pq} is subtracted. Due to the minimization, it is therefore preferable to set \bar{S}_{pq} to a negative value. However, its contribution also needs to exceed the benefit of breaking the synchronization in the remaining part of the objective function.

To ensure the synchronization, we could, in theory, set $\bar{S}_{pq} = -\infty$ for all $pq \in E_\varphi$ or at least to a very large negative value, e.g. exceeding the sum of the absolute values of all coefficients in the embedded Ising model. In this case, we could also choose \bar{W} and \bar{S}_{E_δ} arbitrarily within the sum bounds. However, these large strength values cannot be realized in practice because the annealing machines have a limited parameter precision and height due to physical restrictions. Thus, how do we need to choose the parameters \bar{S}_{E_φ} such that they suffice and how does their choice influence possible choices for \bar{W} and \bar{S}_{E_δ} and vice versa?

4.1.3 Related Work

The baseline for all the work around minor embedding and the corresponding parameter setting was developed by V. Choi. In [12], a first upper bound on the strengths on the inter-coupling edges depending on the original parameters is given, achieved by providing an explicit non-uniform weighting of the vertices in the hardware graph. However, in practice, these bounds seem to be too weak and the large strengths they introduce suppress the success probability due to the necessary scaling factor.

By now, there is a common understanding in the quantum annealing community that the *coupling strength*, the common strength that is in most cases simply applied to all inter-connecting edges, needs to be larger than the largest absolute strength that appears in the original Ising model. Usually, a factor of 2 is applied, as for instance is described in [44]. At the same time, the weights are in general distributed uniformly over the vertices.

Another method used in practice is determining the scaling factor empirically, see e.g. [51]. This means that several instances of the same original problem are transferred into Ising models, usually yielding a common structure. By successively solving the problems with certain parameters and checking the feasibility of the found solutions afterwards, a specific bound or a bounding function in the input parameters is estimated and assumed to hold also for all other instances of the same problem. This however does not provide any provable equivalence of the embedded Ising model.

In the package `dwave-system`, D-Wave's programming interface offers a method to set the coupling strength called 'uniform torque compensation' [15], which is most likely based on [44]. In the given formulation, it only applies for chains, which means if φ_v for $v \in V(G)$ induces a path in the hardware graph. The method is derived from the idea that a 'torque' on the central edge of the chain, caused by the supposedly random influence of the neighboring chains, needs to be compensated by setting the weights and strengths accordingly. As several physical dynamics of the annealing process are included in the considerations at the same time, this approach does not clearly divide the transformation steps towards the machine as we intend to do. Although the results of the empirical study for certain random instances in [44] are promising, an analytical study of the equivalence of the thus obtained solutions is missing, which is why this method can again only be considered as a heuristic approach to obtain the coupling strength.

Due to its physical properties, the quantum machine does only provide solutions with a certain probability. Thus, even if the optimal solution of the embedded Ising problem might be a synchronized one, the machine might miss it in the annealing process and rather provide only suboptimal and unsynchronized solutions. In such cases, the common practice is to apply a post-processing on these unsynchronized solutions. A popular example is the *majority voting*, where the original variable gets the value which appears in the majority of the assignments of the embedded variables [31].

This however changes the contribution of some edges by their strength to the objective value. Thus, broken embeddings might have a global impact on the assignment of a large number of variables, which can usually not be ‘repaired locally’. In particular, applying this method to solutions of an ill-defined embedded Ising problem will not increase the probability of finding the optimal solution. On the other hand, we do not see a way how to construct the embedded Ising such that we obtain the provable equivalence to the original Ising problem under such solutions, due to the large number of possible distributions.

Although the choice of the strength(s) in the single vertex embeddings is decisive for the success probability of the D-Wave machine [51], we are not aware of any further publication that deals with the parameter setting problem of the embedded Ising model analogously or as an extension to V. Choi in [12]. We analyze the problem in the following section and evaluate the resulting optimization problem in detail in the further course of this work.

4.1.4 Single Vertex Evaluation

To answer the question stated at the end of Section 4.1.2, we extract the part of the embedded Ising model that concerns a single original vertex $\mathbf{v} \in V(G)$, which we assume to be fixed in the following:

$$\begin{aligned} \bar{I}(s) = & \sum_{w \in V(G) \setminus \{v\}} \left(\sum_{q \in \varphi_w} \bar{W}_q s_q + \sum_{pq \in E(H[\varphi_w])} \bar{S}_{pq} s_p s_q \right) + \sum_{wu \in E(G) \setminus \delta(v)} \sum_{pq \in \delta_{wu}} \bar{S}_{pq} s_p s_q \\ & + \underbrace{\sum_{q \in \varphi_v} \bar{W}_q s_q + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} s_p s_q + \sum_{w \in N(v)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} s_p s_q}_{=: \bar{I}^v(s)}. \end{aligned}$$

By this the remaining part $\bar{I}^{V(G) \setminus \{v\}}(s) := \bar{I}(s) - \bar{I}^v(s)$ does only depend on $s \in \{-1, 1\}^{V(H_\varphi) \setminus \varphi_v}$. By replacing $s \in \{-1, 1\}^{\varphi_v \cup N(\varphi_v)}$ with $(r, s) \in \{-1, 1\}^{\varphi_v} \times \{-1, 1\}^{N(\varphi_v)}$ in $\bar{I}^v(s)$ we get

$$\bar{I}^v(r, s) = \sum_{q \in \varphi_v} \bar{W}_q r_q + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} r_p r_q + \sum_{w \in N(v)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} s_p r_q$$

and can clearly indicate the different influencing parts. All variables corresponding to the embedding of vertex v , the r -variables, now only appear in \bar{I}^v , while the s -variables form the connection to the remaining part, thus appear in both $\bar{I}^{V(G) \setminus \{v\}}$ and \bar{I}^v .

In the following, we want to enforce the synchronization of the r ’s independently of the influence ‘from the outside’, which means for arbitrary s . Due to the minimization of the Ising models, this means that the minimum of the partial Ising problem should always be either $\mathbb{1}$ or $-\mathbb{1}$, more formally

$$\arg \min_{r \in \{-1, 1\}^{\varphi_v}} \bar{I}^v(r, s) \subseteq \{-1, 1\} \quad \forall s \in \{-1, 1\}^{N(\varphi_v)}.$$

In other words, we have

$$\min_{r \in \{-1, 1\}^{\varphi_v}} \bar{I}^v(r, s) = \min \{ \bar{I}^v(-\mathbb{1}, s), \bar{I}^v(\mathbb{1}, s) \}$$

but with

$$\bar{I}^v(r, s) > \min \{ \bar{I}^v(-\mathbb{1}, s), \bar{I}^v(\mathbb{1}, s) \} \quad \forall s \in \{-1, 1\}^{N(\varphi_v)} \quad \forall r \in \{-1, 1\}^{\varphi_v} \setminus \{-\mathbb{1}, \mathbb{1}\}.$$

Do these conditions ensure that the embedded problem is provably equivalent to the original one? We can indeed show their sufficiency:

Lemma 4.6. *With*

$$\bar{I}^v(r, s) > \min \{ \bar{I}^v(-\mathbb{1}, s), \bar{I}^v(\mathbb{1}, s) \} \quad \forall s \in \{-1, 1\}^{N(\varphi_v)} \quad \forall r \in \{-1, 1\}^{\varphi_v} \setminus \{-\mathbb{1}, \mathbb{1}\}$$

for all $v \in V(G)$, we have for all

$$s^* \in \arg \min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{I}(s)$$

that $s^* = (t_v^* \mathbb{1}_{\varphi_v})_{v \in V(G)}$ with $t^* \in \{-1, 1\}^V$.

Proof. Assume there exists

$$s^* \in \arg \min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{I}(s)$$

with $s_{\varphi_v}^* \notin \{-1, 1\}$ for some vertex $v \in V(G)$. Then we have

$$\begin{aligned} \bar{I}(s^*) &= \bar{I}^v \left(s_{\varphi_v}^*, s_{N(\varphi_v)}^* \right) + \bar{I}^{V(G) \setminus \{v\}} \left(s_{V(H_\varphi) \setminus \varphi_v}^* \right) \\ &> \min \left\{ \bar{I}^v \left(-\mathbb{1}, s_{N(\varphi_v)}^* \right), \bar{I}^v \left(\mathbb{1}, s_{N(\varphi_v)}^* \right) \right\} + \bar{I}^{V(G) \setminus \{v\}} \left(s_{V(H_\varphi) \setminus \varphi_v}^* \right) \end{aligned}$$

by the given conditions and we can further deduce

$$\begin{aligned} \bar{I}(s^*) &= \bar{I}^v \left(r^* \mathbb{1}, s_{N(\varphi_v)}^* \right) + \bar{I}^{V(G) \setminus \{v\}} \left(s_{V(H_\varphi) \setminus \varphi_v}^* \right) \\ &= \bar{I}(\tilde{s}^*) \end{aligned}$$

for $\tilde{s} \in \{-1, 1\}^{V(H_\varphi)}$ with $\tilde{s}_{V(H_\varphi) \setminus \varphi_v} = s_{V(H_\varphi) \setminus \varphi_v}^*$ and $\tilde{s}_{\varphi_v} = r^* \mathbb{1}$ for

$$r^* = \begin{cases} 1 & \text{if } \bar{I}^v \left(-\mathbb{1}, s_{N(\varphi_v)}^* \right) \geq \bar{I}^v \left(\mathbb{1}, s_{N(\varphi_v)}^* \right), \\ -1 & \text{otherwise.} \end{cases}$$

This contradicts to s^* being an optimal solution. □

With this result, we can now clearly formulate the requirements on an embedded Ising model:

Theorem 4.7. For two graphs G and H , an embedding φ of G in H and an Ising model $I_{W,S} : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ with weights $W \in \mathbb{R}^{V(G)}$ and strengths $S \in \mathbb{R}^{E(G)}$, the Ising model $\bar{I}_{\bar{W}, \bar{S}} : \{-1, 1\}^{V(H_\varphi)} \rightarrow \mathbb{R}$ with weights $\bar{W} \in \mathbb{R}^{V(H_\varphi)}$ and strengths $\bar{S} \in \mathbb{R}^{E(H_\varphi)}$ forms an equivalent embedded Ising model to $I_{W,S}$ if we have

$$\begin{aligned} W_v &= \sum_{q \in \varphi_v} \bar{W}_q & \forall v \in V(G), \\ S_{vw} &= \sum_{pq \in \delta_{vw}} \bar{S}_{pq} & \forall vw \in E(G), \\ \bar{I}_{\bar{W}, \bar{S}}^v(r, s) &> \min \left\{ \bar{I}_{\bar{W}, \bar{S}}^v(-\mathbb{1}, s), \bar{I}_{\bar{W}, \bar{S}}^v(\mathbb{1}, s) \right\} & \forall s \in \{-1, 1\}^{N(\varphi_v)} \quad \forall r \in \{-1, 1\}^{\varphi_v} \setminus \{-\mathbb{1}, \mathbb{1}\} \\ & & \forall v \in V(G). \end{aligned}$$

Proof. The optimality is clear with the deductions from the beginning of this section and Lemma 4.6. Furthermore, from an optimal solution

$$s^* \in \arg \min_{s \in \{-1, 1\}^{V(H_\varphi)}} \bar{I}(s),$$

we can easily get a solution of the original Ising problem with $t_v^* = s_q^*$ for an arbitrarily chosen $p \in \varphi_v$ for all $v \in V$ due to the enforced synchronization. \square

Note that this theorem only shows the sufficiency of our derived conditions. However, it does not state anything about the necessity.

4.2 Optimization Problem Formulation

For calculations on the D-Wave machine, it is essential for the user that the encoded problem indeed represents the original problem the user wants to solve. In this section, we extract and simplify the sufficient requirements on the parameters that need to be fulfilled such that the resulting EMBEDDED ISING PROBLEM provably holds equivalent solutions to those of the given problem.

We assume the two graphs \mathbf{G} and \mathbf{H} , the embedding $\varphi : V(G) \rightarrow 2^{V(H)}$ with the corresponding graph structures of Definition 4.1 and an Ising model $\mathbf{I}_{\mathbf{W}, \mathbf{S}} : \{-1, 1\}^{V(G)} \rightarrow \mathbb{R}$ with the weights \mathbf{W} and strengths \mathbf{S} to be given and fixed in the following. Given this data, how do we find an equivalent embedded Ising model $\bar{I}_{\bar{W}, \bar{S}} : \{-1, 1\}^{V(H_\varphi)} \rightarrow \mathbb{R}$ to I with weights \bar{W} and strengths \bar{S} ? Note that we drop the subscripts of the Ising models in most cases for simplicity.

4.2.1 Instance Definition

In the following, we only concentrate on $\bar{I}^v(s)$ for a fixed $v \in V(G)$. From this part of the embedded Ising model, we derive the specific optimization problem that needs to be solved to obtain those parameter values that ensure that the embedded vertices represent the original ones.

Input

The part of the embedded graph \bar{I}^v is working on is the *embedded subgraph structure*

$$H[\varphi_v \cup N(\varphi_v)] = (\varphi_v \cup N(\varphi_v), E(H[\varphi_v]) \cup \delta(\varphi_v) \cup E(H[N(\varphi_v)])),$$

where we have

- the connected inner graph $H[\varphi_v] =: (V, E)$ with vertices \mathbf{V} and edges \mathbf{E} ,
- the outer neighbors $\mathbf{N} := N(\varphi_v) \subseteq \bigcup\{p \in \varphi_w : w \in N(v)\}$,
- the set of edges to the outer neighbors $\mathbf{O} := \delta(\varphi_v)$ and
- the edges between the outer neighbors $E(H[N(\varphi_v)])$.

An example is shown in Figure 4.1. Note that the quadratic terms for the edges between the outer neighbors do not include variables corresponding to vertices in V . Therefore, they are not considered in the definition of \bar{I}^v . In the following section, we nevertheless argue why we can omit these edges.

Although, apart from the constraints

$$S_{vw} = \sum_{pq \in \delta_{vw}} \bar{S}_{pq} \quad \forall w \in N(v),$$

we are free to choose the values for \bar{S}_{pq} for all $pq \in \delta(\varphi_v)$, this introduces another level of complexity to the overall problem. Their choice does not only concern the evaluated vertex v but also its neighbors in G . We keep this additional level for future research and assume in the following that \bar{S}_{pq} is validly chosen in advance and thus given and fixed. Nevertheless, we discuss possible choices supporting the simplification of the problem in the following section and see that our approach can be applied in any case.

All in all, we assume to be given

- the strengths on the outer edges $\bar{\mathbf{S}} \in \mathbb{R}^{\mathbf{O}}$, where we have $\{n \in N : \ell n \in \mathbf{O}, S_n \neq 0\} \neq \emptyset$ for all leaves $\ell \in V$ of G , as they otherwise could be ‘cut off’ from the embedding, and
- the total weight $\mathbf{W} := W_v \in \mathbb{R}$.

All of the objects marked in bold are assumed to be fixed in the following.

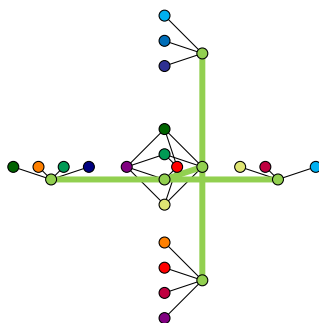


Figure 4.1: Example for an embedded subgraph structure of a single vertex with all outer neighbors extracted from the complete graph embedding in the broken Chimera graph of Figure 3.22(b)

Besides, by excluding possibly existing additional inner edges, G could be reduced to a tree. We use this fact later on, as several problems are much easier on trees. However, for the first steps, we consider G to be an arbitrary graph.

Output and Objective

With the notations and simplifications of the previous section, we have

$$\begin{aligned}\bar{I}^v(s) &= \sum_{q \in \varphi_v} \bar{W}_q s_q + \sum_{pq \in E(H[\varphi_v])} \bar{S}_{pq} s_p s_q + \sum_{w \in N(v)} \sum_{pq \in \delta_{vw}} \bar{S}_{pq} s_p s_q \\ &= \sum_{q \in V} \bar{W}_q s_q + \sum_{pq \in E} \bar{S}_{pq} s_p s_q + \sum_{q \in V} \sum_{qn \in O} \bar{S}_{qn} s_q s_n\end{aligned}$$

and further get by analogously replacing $s \in \{-1, 1\}^{\varphi_v \cup N(\varphi_v)}$ with $(r, s) \in \{-1, 1\}^V \times \{-1, 1\}^N$, as shown in the previous section, and renaming the parameters

$$\begin{aligned}\bar{I}^v(s) &= \sum_{q \in V} \omega_q r_q - \sum_{pq \in E} \beta_{pq} r_p r_q + \sum_{q \in V} \sum_{qn \in O} \bar{S}_{qn} s_n r_q \\ &=: I_{\omega, \beta}^{\bar{S}}(\mathbf{r}, \mathbf{s}),\end{aligned}$$

the Ising model $I_{\omega, \beta}^{\bar{S}} : \{-1, 1\}^V \times \{-1, 1\}^N \rightarrow \mathbb{R}$, where we drop the bar and the superscript v for simplicity. We now search for the parameters:

- the weights $\omega := \bar{W}_{\varphi_v} \in \mathbb{R}^V$ and
- the strengths $\beta := -\bar{S}_{E(H[\varphi_v])} \in \mathbb{R}^E$.

The larger we choose β_{pq} , the stronger the vertices p and q are coupled due to the negative sign in $I_{\omega, \beta}^{\bar{S}}$. As discussed in Section 2.2, we cannot simply set these strengths to some very large value compared to the remaining parameters due to the machine restrictions. In literature, the coupling strength is mentioned to be decisive for the success probability, however usually yielding the maximal absolute coefficient C_{\max} at the same time. Therefore, the question arises: How small can we set these strengths such that we can still achieve an equivalent embedded Ising? This means that a first step based on current practice would be to simply minimize $\|\beta\|_{\infty}$.

However, by only minimizing the strengths, a corresponding suitable weighting could exceed the corresponding bound in some vertices. Hence, with the strengths on the outer edges assumed to be fixed, the more interesting objective would be the maximal absolute value of all remaining parameters of the observed part of the Ising model $\max\{\|\omega\|_{\infty}, \|\beta\|_{\infty}\}$, which should be minimized in total.

Constraints

In most of the following work, we do not refer to the original vertex v anymore but only deal with a single embedded graph structure and the corresponding Ising model as defined before. Thus, we simply use $v \in V$ to refer to a vertex in the hardware belonging to the inner graph of the embedded subgraph structure. By the previous section, we can already derive the following constraints on the introduced parameters ω and β : The weights should sum up to the total weight with

$$W = \sum_{v \in V} \omega_v =: \omega(V)$$

and, from the conditions of Lemma 4.6, we need

$$I_{\omega,\beta}^{\bar{S}}(r, s) > \min \left\{ I_{\omega,\beta}^{\bar{S}}(-\mathbb{1}, s), I_{\omega,\beta}^{\bar{S}}(\mathbb{1}, s) \right\} \quad \forall s \in \{-1, 1\}^N \quad \forall r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}, \quad (4.2)$$

to ensure that the full embedded problem is provably equivalent to original one in the end.

Note that the latter condition is comprised of an exponential number of constraints, more precisely $2^{|N|}(2^{|V|}-2)$ many. Although they are linear inequalities, the overall optimization problem is therefore not solvable in polynomial time in a straightforward way. Thus, it could only be used for small graph instances G in practice.

By introducing a *gap* value $\gamma \in \mathbb{R}_{>0}$, we can further influence how ‘far away’, in terms of the distance of their objective values, invalid variable assignments are from the valid ones. This value might become important for the user of the D-Wave machine, when trying to improve the success probability of finding an optimal solution. By this we can also relax the order relation to a greater or equal:

$$I_{\omega,\beta}^{\bar{S}}(r, s) \geq \min \left\{ I_{\omega,\beta}^{\bar{S}}(-\mathbb{1}, s), I_{\omega,\beta}^{\bar{S}}(\mathbb{1}, s) \right\} + \gamma \quad \forall s \in \{-1, 1\}^N \quad \forall r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}.$$

In the following, we assume this value is given with the input and fixed. In future research, we might also investigate different approaches in trading off the gap against the strength. This however shall not be part of this work.

Assuming the given original Ising model does only contain integer coefficients, we also have $W \in \mathbb{Z}$ and $\bar{S} \in \mathbb{Z}^O$. Considering the limited machine precision, an interesting step is to also restrict the parameters to integers, with $\omega \in \mathbb{Z}^V$ and $\beta \in \mathbb{Z}^E$, to ensure a minimum distance of unequal values of 1. This is an option which might introduce another level of complexity, as it converts the formerly continuous program into an integer one. We briefly investigate its consequences at the end of this chapter. For the start we work with a continuous program.

In a Nutshell

By the previous notations and reformulations, we can now summarize the problem with the two different objective functions, which we define both in one, for shortness. Without the terms in the square brackets, we have the ‘full’ problem version. In the ‘strength-only’ variant, the objective function $\max \{ \|\omega\|_\infty, \|\beta\|_\infty \}$ is replaced by the simpler $\|\beta\|_\infty$.

[STRENGTHS-ONLY] GAPPED PARAMETER SETTING PROBLEM. Given an embedded subgraph $(V \cup N, E \cup O)$, $\bar{S} \in \mathbb{R}^O$, $W \in \mathbb{R}$ and $\gamma \in \mathbb{R}_{>0}$, find ω and β that solve

$$\begin{aligned} \min \max \{ & \|\omega\|_\infty, \|\beta\|_\infty \} & & [\|\beta\|_\infty] \\ \text{s.t. } & \omega \in \mathbb{R}^V, \beta \in \mathbb{R}^E, \\ & \omega(V) = W, \\ & I_{\omega,\beta}^{\bar{S}}(r, s) \geq \min \left\{ I_{\omega,\beta}^{\bar{S}}(-\mathbb{1}, s), I_{\omega,\beta}^{\bar{S}}(\mathbb{1}, s) \right\} + \gamma \quad \forall s \in \{-1, 1\}^N \\ & & & \forall r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}. \end{aligned}$$

4.2.2 Simplifications

The instance defined in the previous section can be simplified due to some properties of the Ising models. We can apply several steps, which are discussed in the following.

Common Strength

The β -variables only introduce an offset for synchronized variable assignments. As the total size of this offset is irrelevant, we can choose to set all strengths to the same maximal value. Thus, with $\vartheta := \|\beta\|_\infty$, we can set β_{vw} to ϑ for all $vw \in E$ and therefore get

$$I_{\omega, \vartheta}^{\bar{S}}(r, s) = \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sum_{vn \in O} \bar{S}_{vn} s_n r_v. \quad (4.3)$$

This means that the objective function to be minimized in the simplified version is only ϑ .

For the full version, instead of using $\max\{\|\omega\|_\infty, \vartheta\}$ directly as an objective function, we can further simplify the formulation by introducing a further constraint: With $\vartheta \geq \|\omega\|_\infty$, we can still minimize ϑ . The constant c describing the difference to the original Ising model is now simply $\vartheta|E|$.

Non-negative Input Parameters Due to Symmetry

Furthermore, we can take advantage of the symmetry in the Ising model being defined over variables in $\{-1, 1\}$. By replacing s with $\bar{s} = -s$, we can switch the sign of the strengths on the intra-coupling edges:

$$\begin{aligned} I_{\omega, \vartheta}^{\bar{S}}(r, s) &= \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sum_{vn \in O} \bar{S}_{vn} s_n r_v \\ &= \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sum_{vn \in O} (-\bar{S}_{vn}) \bar{s}_n r_v \\ &= I_{\omega, \vartheta}^{-\bar{S}}(r, \bar{s}). \end{aligned}$$

We can see that the different assignments of the s -variables in (4.2) cover all possible sign combinations of the strengths \bar{S} . Thus, we can restrict \bar{S} to $\mathbb{R}_{\geq 0}^O$ in the following evaluations.

Additionally, by replacing (r, s) with (\bar{r}, \bar{s}) , we observe a symmetry for the ω 's: We have

$$\begin{aligned} I_{\omega, \vartheta}^{\bar{S}}(r, s) &= \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sum_{vn \in O} \bar{S}_{vn} s_n r_v \\ &= \sum_{v \in V} (-\omega_v) \bar{r}_v - \vartheta \sum_{vw \in E} \bar{r}_v \bar{r}_w + \sum_{v \in V} \sum_{vn \in O} \bar{S}_{vn} \bar{s}_n \bar{r}_v \\ &= I_{-\omega, \vartheta}^{\bar{S}}(\bar{r}, \bar{s}) \end{aligned}$$

with

$$\sum_{v \in V} (-\omega_v) = -W.$$

This means we can analogously restrict the total weight to $W \in \mathbb{R}_{\geq 0}$.

Independent Outer Neighbors

By the embedding definition, a single edge connecting the embeddings of different vertices is sufficient. In practice, we usually have $|\delta_{vw}| > 1$ for at least a few pairs of original vertices vw due to the symmetric structure of the Chimera. This can also be seen in Figure 4.1. Here, we discuss some options and the assumptions which we base the following work on.

If there are multiple edges between the embeddings of two vertices, one possibility could simply be to choose a certain edge $\bar{e} \in \delta_{vw}$ and ‘ignore’ the others. This means, for the embedded Ising model, we could set

$$\begin{aligned}\bar{S}_{\bar{e}} &= S_{vw}, \\ \bar{S}_e &= 0 \quad \forall e \in \delta_{vw} \setminus \{\bar{e}\}\end{aligned}$$

and further only deal with $\delta_{vw} = \{\bar{e}\}$. This would influence the former graph structure in two ways:

1. No two vertices of V share an outer neighbor, which means we have $|\{q : qn \in \delta(\varphi_v)\}| = 1$ for all $n \in N$ and thus $O = \delta(\varphi_v) \cong N$.
2. We have $E(H[N(\varphi_v)]) = \emptyset$, which means that no two outer neighbors are connected by an edge.

By this, on the one hand, we have already achieved a simplification of the problem and, on the other hand, we have ensured that all the outer neighbors are independent of each other because they belong to different original neighbors of v .

However, another possibility also offers an advantage: By spreading the strength equally over all of the available edges as suggested in [44] with

$$\bar{S}_{pq} = \frac{S_{vw}}{|\delta_{vw}|} \quad \forall pq \in \delta_{vw},$$

the coefficients are decreased. This seems to be beneficial for complying with the parameter range of the machine. In case of $S \in \mathbb{Z}^{E(G)}$, we can also keep $\bar{S}_{\delta_{vw}} \in \mathbb{Z}^{\delta_{vw}}$ for all $vw \in E(G)$ by simple rounding. As the variables corresponding to the embeddings of the other vertices shall be synchronized, too, the outer neighbors of some inner vertices are not independent of each other in this case. While in the Chimera graph no triangles exist, they might be present in other hardware graphs, such as the Pegasus graph, and might cause inner vertices that even share a common outer neighbor.

Howsoever the strengths are distributed, we want to take advantage of both above strategies. Thus, we simplify the embedded graph structure by splitting up possibly existing shared vertices and simply considering all outer vertices to be pairwise independent of each other as if they would belong to different neighbors of the original vertex. Thus, for the following considerations in this work, we assume that we have $O \cong N$. With this we now consider $\mathbf{S} \in \mathbb{R}_{\geq 0}^N$ to be the given and fixed strengths, yielding the points 1 and 2 from above.

These assumptions become relevant in the following section, where we estimate the worst cases of the ‘outer influence’, the contribution of the strengths on the edges to the outer neighbors multiplied by the s -variables, to further simplify \bar{I}^v . If there exist two variables s_n and s_m corresponding to two outer neighbors $n, m \in N$ that belong to the same embedding φ_w of an original neighboring vertex w , those variables would get the same value in a synchronized solution of the overall problem. As we however assume they can be chosen arbitrarily and independently, the worst case might be overestimated. This means our derived bounds still hold, but the found

strength might be larger than actually necessary in this case. Additionally, the actual minimal gap between a synchronized and a unsynchronized solution might be larger than intended by the user.

Single Outer Neighbor

Finally, we further assume that the given graph instance does only have a single outer neighbor for each vertex $v \in V$. This means we have $N \cong V$. If this is not the case, which means there are at least two outer neighbors for a single inner vertex, the following lemma shows that a synchronization of the variables over these outer neighbors suffices to also cover those cases where the strengths on the outer edges get different signs.

Lemma 4.8. For $I_{\omega, \vartheta}^S : \{-1, 1\}^V \times \{-1, 1\}^N$ as given in (4.3) with $S \in \mathbb{R}_{\geq 0}^N$, the gap value $\gamma \in \mathbb{R}_{\geq 0}$, a vertex $v \in V$ and its neighbors $n, m \in N$ with $vn, vm \in O$, we have

$$\begin{aligned} I_{\omega, \vartheta}^S(r, s) &\geq \min \{I_{\omega, \vartheta}^S(-\mathbb{1}, s), I_{\omega, \vartheta}^S(\mathbb{1}, s)\} + \gamma \quad \text{for } s_n = s_m \in \{-1, 1\} \\ \Rightarrow I_{\omega, \vartheta}^S(r, s) &\geq \min \{I_{\omega, \vartheta}^S(-\mathbb{1}, s), I_{\omega, \vartheta}^S(\mathbb{1}, s)\} + \gamma \quad \text{for } s_n, s_m \in \{-1, 1\} \end{aligned}$$

for all $r \in \{-1, 1\}^V$ and all $s_{N \setminus \{m, n\}} \in \{-1, 1\}^{N \setminus \{m, n\}}$.

Proof. In the following proof, we drop the sub- and superscripts on I for simplicity. With

$$\begin{aligned} I(r, s) &= S_n s_n r_v + S_m s_m r_v - \vartheta |E| + \gamma + r_v \underbrace{\left(\omega_v - \vartheta \sum_{vw \in E} r_w + \sum_{\substack{v\ell \in O \\ \ell \neq n, m}} S_\ell s_\ell \right)}_{=: A(r, s)} \\ &\quad + \vartheta |E| - \gamma + \underbrace{\sum_{w \in V \setminus \{v\}} \omega_w r_w - \vartheta \sum_{\substack{wu \in E \\ w, u \neq v}} r_w r_u + \sum_{w \in V \setminus \{v\}} \sum_{w\ell \in O} S_\ell s_\ell r_w}_{=: B(r, s)} \end{aligned}$$

and

$$I(\pm \mathbb{1}, s) = \pm \left(S_n s_n + S_m s_m + \underbrace{\sum_{\substack{\ell \in N \\ \ell \neq n, m}} S_\ell s_\ell + W}_{=: C(r, s)} \right) - \vartheta |E| + \gamma,$$

the functions A , B and C are independent of r_v , s_n and s_m . By the condition of the lemma, we know

$$\begin{aligned} S_n s_n r_v + S_m s_m r_v + A(r, s) r_v + B(r, s) &\geq -|S_n s_n + S_m s_m + C(r, s)| \\ \forall s_n = s_m \in \{-1, 1\} \quad \forall r_v \in \{-1, 1\} \quad \forall s \in \{-1, 1\}^{N \setminus \{n, m\}} \quad \forall r \in \{-1, 1\}^{V \setminus \{v\}} \setminus \{-1, 1\} \end{aligned}$$

and it remains to show that from this follows

$$\begin{aligned} S_n s_n r_v + S_m s_m r_v + A(r, s) r_v + B(r, s) &\geq -|S_n s_n + S_m s_m + C(r, s)| \\ \forall s_n \neq s_m \in \{-1, 1\} \quad \forall r_v \in \{-1, 1\} \quad \forall s \in \{-1, 1\}^{N \setminus \{n, m\}} \quad \forall r \in \{-1, 1\}^{V \setminus \{v\}} \setminus \{-1, 1\}. \end{aligned}$$

More simply, this means to show that, for any fixed $A, B, C \in \mathbb{R}$ and $S_1, S_2 \in \mathbb{R}_{\geq 0}$ with

$$(S_1 + S_2)sr + Ar + B \geq -|(S_1 + S_2)s + C| \quad \forall r, s \in \{-1, 1\}, \quad (\text{i})$$

it follows that

$$(S_1 - S_2)sr + Ar + B \geq -|(S_1 - S_2)s + C| \quad \forall r, s \in \{-1, 1\}. \quad (\text{ii})$$

This can be achieved by the following two case distinctions. By inserting $s = -1$ and $r = 1$ in inequality (i), we obtain due to $S_1, S_2 \geq 0$

- for $C \leq S_1 + S_2$

$$\begin{aligned} & -S_1 - S_2 + A + B \geq -|(S_1 + S_2) + C| = -S_1 - S_2 + C \\ \Leftrightarrow & \quad A + B \geq C \\ \Leftrightarrow & (S_1 - S_2)s + A + B \geq (S_1 - S_2)s + C \\ & \geq -|(S_1 - S_2)s + C| \quad \forall s \in \{-1, 1\} \end{aligned}$$

- and for $C > S_1 + S_2$

$$\begin{aligned} & -S_1 - S_2 + A + B \geq -|(S_1 + S_2) + C| = S_1 + S_2 - C \\ \Rightarrow & (S_1 - S_2)s + A + B \geq -S_1 - S_2 + A + B \\ & \geq S_1 + S_2 - C \\ & \geq -(S_1 - S_2)s - C \\ & \geq -|(S_1 - S_2)s + C| \quad \forall s \in \{-1, 1\}. \end{aligned}$$

Thus, we have shown the implication of inequality (ii) for $r = 1$ and arbitrary s . By inserting $s = 1$ and $r = -1$ in inequality (i), we obtain in turn

- for $C \geq -S_1 - S_2$

$$\begin{aligned} & -S_1 - S_2 - A + B \geq -|S_1 + S_2 + C| = -S_1 - S_2 - C \\ \Leftrightarrow & \quad -A + B \geq -C \\ \Leftrightarrow & -(S_1 - S_2)s - A + B > -(S_1 - S_2)s - C \\ & \geq -|-(S_1 - S_2)s - C| \\ & = -|(S_1 - S_2)s + C| \quad \forall s \in \{-1, 1\} \end{aligned}$$

- and for $C < -S_1 - S_2$

$$\begin{aligned} & -S_1 - S_2 - A + B > -|S_1 + S_2 + C| = S_1 + S_2 + C \\ \Rightarrow & -(S_1 - S_2)s - A + B \geq -S_1 - S_2 - A + B \\ & \geq S_1 + S_2 + C \\ & \geq (S_1 - S_2)s + C \\ & \geq -|(S_1 - S_2)s + C| \quad \forall s \in \{-1, 1\}. \end{aligned}$$

This means we have also shown implication of inequality (ii) for $r = -1$ and arbitrary s , which completes the proof. \square

With the conditions of the lemma, we can replace the single occurrence of s_m with s_n in $I_{\omega, \vartheta}^S(r, s)$. This way s_n gets the coefficient $S_n + S_m$ and we have reduced the outer neighbors by 1 by removing m . We can apply this lemma iteratively to all outer neighbors that share an inner vertex. Hence, by the lemma, we can say that the synchronization of the variables corresponding

to the outer neighbors forms the ‘worst case’ with respect to the outer influence. By defining the weighting $\sigma \in \mathbb{R}_{\geq 0}^V$ with

$$\sigma_v := \sum_{vn \in O} S_v \quad \forall v \in V,$$

we can therefore reduce the Ising model to

$$I_{\omega, \vartheta}^{\sigma}(\mathbf{r}, \mathbf{s}) = \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sigma_v s_v r_v,$$

where we reindexed the s -variables due to the isomorphism of V and N . Thus, only $\sigma \in \mathbb{R}_{\geq 0}^V$ is considered in the following as an input for the weight distribution problem and we do not require the outer neighbors N and the edges O to them anymore.

Graph Cuts

With the former simplifications, we can now further evaluate the condition

$$I_{\omega, \vartheta}^{\sigma}(r, s) \geq \min \{I_{\omega, \vartheta}^{\sigma}(-\mathbb{1}, s), I_{\omega, \vartheta}^{\sigma}(\mathbb{1}, s)\} + \gamma \quad \forall s \in \{-1, 1\}^V \quad \forall r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}$$

for fixed parameters and reformulate it using graph cuts. Inserting the full formulation for $I_{\omega, \vartheta}^{\sigma}$, we have for all $s \in \{-1, 1\}^V$ and for all $r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}$:

$$\begin{aligned} & \sum_{v \in V} \omega_v r_v - \vartheta \sum_{vw \in E} r_v r_w + \sum_{v \in V} \sigma_v s_v r_v \\ \geq & \min \left\{ \sum_{v \in V} \omega_v (-1) - \vartheta \sum_{vw \in E} (-1)(-1) + \sum_{v \in V} \sigma_v s_v (-1), \sum_{v \in V} \omega_v - \vartheta \sum_{vw \in E} 1 + \sum_{v \in V} \sigma_v s_v \right\} + \gamma \\ = & \min \left\{ - \sum_{v \in V} (\sigma_v s_v + \omega_v), \sum_{v \in V} (\sigma_v s_v + \omega_v) \right\} - \vartheta |E| + \gamma, \end{aligned}$$

which is equivalent to

$$\begin{aligned} \vartheta |E| - \vartheta \sum_{vw \in E} r_v r_w &= \vartheta \left(|E| - \sum_{vw \in E} r_v r_w \right) \\ &= \vartheta (|E| - |\{vw \in E : r_v = r_w\}| + |\{vw \in E : r_v = -r_w\}|) \\ &= 2\vartheta |\{vw \in E : r_v = -r_w\}| \\ &\geq \min \left\{ \sum_{v \in V} (\sigma_v s_v + \omega_v), - \sum_{v \in V} (\sigma_v s_v + \omega_v) \right\} - \sum_{v \in V} (\sigma_v s_v + \omega_v) r_v + \gamma \\ &= \min \left\{ \sum_{v \in V} (\sigma_v s_v + \omega_v)(1 - r_v), \sum_{v \in V} (\sigma_v s_v + \omega_v)(-1 - r_v) \right\} + \gamma \\ &= 2 \min \left\{ \sum_{\substack{v \in V \\ r_v = -1}} (\sigma_v s_v + \omega_v), - \sum_{\substack{v \in V \\ r_v = 1}} (\sigma_v s_v + \omega_v) \right\} + \gamma. \end{aligned}$$

By combining all the lower bounds, we can further deduce stronger conditions, where we have for all $r \in \{-1, 1\}^V \setminus \{-\mathbb{1}, \mathbb{1}\}$:

$$\begin{aligned} \vartheta|\{vw \in E : r_v = -r_w\}| &\geq \max_{s \in \{-1, 1\}^V} \min \left\{ \sum_{\substack{v \in V \\ r_v = -1}} (\sigma_v s_v + \omega_v), - \sum_{\substack{v \in V \\ r_v = 1}} (\sigma_v s_v + \omega_v) \right\} + \frac{1}{2}\gamma \\ &= \min \left\{ \sum_{\substack{v \in V \\ r_v = -1}} (\sigma_v + \omega_v), \sum_{\substack{v \in V \\ r_v = 1}} (\sigma_v - \omega_v) \right\} + \frac{1}{2}\gamma. \end{aligned}$$

It is easy to recognize that the maximum in the former relation is achieved when setting s_v to $-r_v$ for all $v \in V$, leading to the last equality. Note that we already reduced the number of constraints to $2^{|V|} - 2$ with this step.

With the vertex set definition of $S = \{v \in V : r_v = -1\}$ for a specific assignment of the r -variables and

$$\sigma(S) := \sum_{v \in S} \sigma_v,$$

we can equivalently formulate for all $\emptyset \neq S \subsetneq V$:

$$\begin{aligned} \vartheta|\{vw \in E : v \in S, w \in V \setminus S\}| = \vartheta|\delta(S)| &\geq \min \left\{ \sum_{v \in S} (\sigma_v + \omega_v), \sum_{v \in V \setminus S} (\sigma_v - \omega_v) \right\} + \frac{1}{2}\gamma \\ &= \min \{ \sigma(S) + \omega(S), \sigma(V \setminus S) - \omega(V \setminus S) \} + \frac{1}{2}\gamma. \end{aligned}$$

Note that the empty set and the full vertex set are excluded because the r -variables cannot get all the value 1 or all the value -1 . As furthermore the graph G is connected by the embedding definition, we have $|\delta(S)| > 0$ for all non-trivial cuts S . As ϑ is the objective function, we can reformulate the conditions to lower bounds on the optimal value with

$$\vartheta \geq \frac{\min\{\sigma(S) + \omega(S), \sigma(V \setminus S) - \omega(V \setminus S)\} + \frac{1}{2}\gamma}{|\delta(S)|} \quad \forall \emptyset \neq S \subsetneq V.$$

We refer to them as *cut inequalities* in the following. They can equivalently be combined all in one with

$$\vartheta \geq \max_{\emptyset \neq S \subsetneq V} \left\{ \frac{\min\{\sigma(S) + \omega(S), \sigma(V \setminus S) - \omega(V \setminus S)\} + \frac{1}{2}\gamma}{|\delta(S)|} \right\}.$$

Note that, according to the preprocessing step based on Lemma 2.2 of Section 2.2, there are instances for which the weight of the original vertex predominates all outer influences caused by its incident edges. The variable corresponding to such a vertex can be preprocessed before the embedding anyway by setting it according to the sign of the weight. This case appears if we have $W \geq \sigma(V)$. Thus, we also only consider instances with $\sigma(V) > W$ in the following. Finally, we can summarize the full problems in the next section.

Summary

In this section, we briefly summarize the problems that are handled throughout the remaining part of this work derived from the [STRENGTHS-ONLY] GAPPED PARAMETER SETTING PROBLEM. Previously, we have considered two different objectives concerning the strength and the weights. Both have been reduced to simply optimize ϑ while distributing the weights ω . Therefore, we derive the following two problems differing in one constraint that is added in the full compared to the strength-only formulation:

STRENGTH-ONLY [FULL] GAPPED WEIGHT DISTRIBUTION PROBLEM. Given a graph $G = (V, E)$, $\sigma \in \mathbb{R}_{\geq 0}^V$, $W \in \mathbb{R}_{\geq 0}$ with $W < \sigma(V)$ and $\gamma \in \mathbb{R}_{> 0}$, find ϑ and ω that solve

$$\begin{aligned} & \min \vartheta \\ & \text{s.t. } \vartheta \in \mathbb{R}, \omega \in \mathbb{R}^V, \\ & \vartheta \geq \frac{\min \{ \sigma(S) + \omega(S), \sigma(V \setminus S) - \omega(V \setminus S) \} + \gamma}{|\delta(S)|} \quad \forall \emptyset \neq S \subsetneq V, \\ & \omega(V) = W, \\ & [\vartheta \geq \|\omega\|_\infty]. \end{aligned}$$

We can easily see the following relation between the two problems:

Corollary 4.9. *If we have $\vartheta^* \geq \|\omega^*\|_\infty$ for any optimal solution (ϑ^*, ω^*) of the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM, it is also an optimal solution of the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM.*

Let ϑ_δ^γ be the optimal value of the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM and ϑ_W^γ of the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM. We also clearly have $\vartheta_\delta^\gamma \geq \vartheta_W^\gamma$ in general.

We further add a special case here for completeness: In case we have $W = 0$, the ω -variables shall sum up to 0. An apparent reduction step is therefore to omit the ω 's in this case by setting them to 0 in advance. This results in the problem

GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM. Given graph $G = (V, E)$, $\sigma \in \mathbb{R}_{\geq 0}^V$ and $\gamma \in \mathbb{R}_{> 0}$, find ϑ that solves

$$\begin{aligned} & \min \vartheta \\ & \text{s.t. } \vartheta \geq \frac{\min \{ \sigma(S), \sigma(V \setminus S) \} + \gamma}{|\delta(S)|} \quad \forall \emptyset \neq S \subsetneq V, \\ & \vartheta \in \mathbb{R}. \end{aligned}$$

Actually, we indeed show in the following section that distributing a certain positive weight over some vertices and the negative counterpart over the other vertices is not beneficial over setting $\omega = \mathbb{0}$. Accordingly, we can reuse the above notation and denote with ϑ_δ^γ the optimal solution of the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM.

Note that we have substituted $\frac{1}{2}\gamma$ with γ in all problems for simplicity. This means that the actual distance between a synchronized and a non-synchronized solution is 2γ in the end. We further use the term ‘gapped’ here for the problems, in combination with the γ -superscript on the notation of the objective values, to distinguish those objects from the ‘gapless’ versions, which we introduce later on.

4.2.3 LP Formulation

In the following, let the graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, the strengths $\boldsymbol{\sigma} \in \mathbb{R}_{\geq 0}^V$, the total weight $\mathbf{W} \in \mathbb{R}_{\geq 0}$ with $W < \sigma(V)$ and the gap $\boldsymbol{\gamma} \in \mathbb{R}_{> 0}$ be given and fixed. For $\emptyset \neq S_1 \subseteq S_2 \subseteq V$, we see the σ -sums are monotonic over the partial ordering of the subset relation with

$$0 \leq \sigma(S_1) \leq \sigma(S_2) \leq \sigma(V)$$

due to $\sigma \geq \mathbb{O}$. With

$$\begin{aligned} \sigma(S) + \omega(S) &\stackrel{\leq}{\leq} \sigma(V \setminus S) - \omega(V \setminus S) \\ \Leftrightarrow \sigma(S) &\stackrel{\leq}{\leq} \frac{1}{2} (\sigma(V) - W) \\ \Leftrightarrow \sigma(V \setminus S) &\stackrel{\leq}{\leq} \frac{1}{2} (\sigma(V) + W) \\ \Leftrightarrow W &\stackrel{\leq}{\leq} \sigma(V \setminus S) - \sigma(S) \end{aligned}$$

and

$$\sigma(V) + W \geq \sigma(V) \geq \sigma(V) - W$$

for $W \geq 0$, we have for all $\emptyset \neq S \subsetneq V$:

$$\vartheta \geq \begin{cases} \frac{\sigma(S) + \omega(S) + \gamma}{|\delta(S)|} & \text{if } \sigma(S) < \frac{1}{2} (\sigma(V) - W), \\ \frac{\sigma(V \setminus S) - \omega(V \setminus S) + \gamma}{|\delta(S)|} & \text{otherwise.} \end{cases}$$

This can be used for the following polyhedra definitions:

Definition 4.10. Let

$$\begin{aligned} \vartheta_{\frac{1}{2}} &:= \frac{1}{2} (\sigma(V) - W), \\ \Theta^\gamma &:= \left\{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \right. \\ &\quad \left. \vartheta \geq \begin{cases} \frac{\sigma(S) + \omega(S) + \gamma}{|\delta(S)|} & \text{if } \sigma(S) < \vartheta_{\frac{1}{2}}, \\ \frac{\sigma(V \setminus S) - \omega(V \setminus S) + \gamma}{|\delta(S)|} & \text{otherwise,} \end{cases} \quad \forall \emptyset \neq S \subsetneq V, \right. \\ &\quad \left. \omega(V) = W \right\} \end{aligned}$$

and

$$\Phi := \{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta \geq \|\omega\|_\infty \}.$$

Note that, with $W < \sigma(V)$, we always have $\vartheta_{\frac{1}{2}} > 0$.

With these definitions, we can easily see that Θ^γ is an unbounded, $|V|$ -dimensional polyhedron described by an exponential number of inequalities. The **STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM** can now also be written as the LP

$$\boxed{\min \{ \vartheta : (\vartheta, \omega) \in \Theta^\gamma \} = \min \{ (1, \mathbb{O})^\top \lambda : \lambda \in \Theta^\gamma \}.$$

The domain of the **FULL GAPPED WEIGHT DISTRIBUTION PROBLEM** is then the intersection of Θ^γ with the cone Φ , which is defined by only $2|V|$ constraints since $\vartheta \geq \|\omega\|_\infty$ is equivalent to

$$\vartheta \geq |\omega_v| \quad \forall v \in V,$$

or without any absolute value also to

$$\begin{aligned}\vartheta &\geq \omega_v & \forall v \in V, \\ \vartheta &\geq -\omega_v & \forall v \in V.\end{aligned}$$

The corresponding LP can be formulated as

$$\min \{ \vartheta : (\vartheta, \omega) \in \Theta^\gamma \cap \Phi \} = \min \{ (1, \mathbf{0})^\top \lambda : \lambda \in \Theta^\gamma \cap \Phi \}.$$

Furthermore, we see that we have one inequality per vertex set. We can reduce the number of vertex sets that need to be considered by combining the inequalities for S and $V \setminus S$ allowing for a more compact notation. Thus, we define:

Definition 4.11. Let

$$\begin{aligned}\mathfrak{S}_1 &:= \{ \emptyset \neq S \subsetneq V : \sigma(S) < \vartheta_{\frac{1}{2}} \}, \\ \mathfrak{S}_2 &:= \{ \emptyset \neq S \subsetneq V : \vartheta_{\frac{1}{2}} \leq \sigma(S) \leq \frac{1}{2}\sigma(V) \}, \\ \Theta^\gamma(S) &:= \begin{cases} \{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta |\delta(S)| \geq \sigma(S) + |\omega(S)| + \gamma \} & \text{if } S \in \mathfrak{S}_1, \\ \{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta |\delta(S)| \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| + \gamma \} & \text{otherwise} \end{cases}\end{aligned}$$

and

$$\Theta_W := \{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \omega(V) = W \}.$$

Remarks:

- The set $\mathfrak{S}_1 \cup \mathfrak{S}_2 = \{ \emptyset \neq S \subsetneq V : \sigma(S) \leq \frac{1}{2}\sigma(V) \}$ contains at least half of all vertex subsets. It contains more only if there exists $\emptyset \neq S \subsetneq V$ with $\sigma(S) = \frac{1}{2}\sigma(V)$. In this case, we have $S, V \setminus S \in \mathfrak{S}_2$.
- The set \mathfrak{S}_1 is closed under taking subsets because, from $S \in \mathfrak{S}_1$ and $\tilde{S} \subseteq S$, it follows that $\sigma(\tilde{S}) \leq \sigma(S) < \vartheta_{\frac{1}{2}}$ and thus $\tilde{S} \in \mathfrak{S}_1$.
- For $S \in \mathfrak{S}_2$ and $\tilde{S} \subseteq S$, we have $\sigma(\tilde{S}) \leq \sigma(S) \leq \frac{1}{2}\sigma(V)$ and therefore $\tilde{S} \in \mathfrak{S}$. Here, we can only deduce $\tilde{S} \in \mathfrak{S}_1$ or $\tilde{S} \in \mathfrak{S}_2$.
- For each $\emptyset \neq S \subsetneq V$, we have $S \in \mathfrak{S}_1, S \in \mathfrak{S}_2, V \setminus S \in \mathfrak{S}_1$ or $V \setminus S \in \mathfrak{S}_2$.

Using these definitions, we can derive a different formulation for the Θ -polyhedron:

Lemma 4.12. *We have*

$$\Theta^\gamma = \bigcap_{S \in \mathfrak{S}_1 \cup \mathfrak{S}_2} \Theta^\gamma(S) \cap \Theta_W.$$

Proof. Considering one set S and its complement $V \setminus S$, w.l.o.g. $\sigma(S) \leq \frac{1}{2}\sigma(V) \leq \sigma(V \setminus S)$, we have two cases:

A) $S \in \mathfrak{S}_1$ with $\sigma(S) < \frac{1}{2}(\sigma(V) - W) = \vartheta_{\frac{1}{2}} \Leftrightarrow \sigma(V \setminus S) > \frac{1}{2}(\sigma(V) + W)$, thus $\sigma(V \setminus S) > \vartheta_{\frac{1}{2}}$, where the constraints generated by S and $V \setminus S$ in the definition of Θ^γ are

$$\begin{aligned} \vartheta|\delta(S)| &\geq \sigma(S) + \omega(S) + \gamma, \\ \vartheta|\delta(S)| &\geq \sigma(V \setminus (V \setminus S)) - \omega(V \setminus (V \setminus S)) + \gamma \\ &= \sigma(S) - \omega(S) + \gamma, \end{aligned}$$

which is equivalent to

$$\begin{aligned} \vartheta|\delta(S)| &\geq \sigma(S) + \max\{\omega(S), -\omega(S)\} + \gamma \\ &= \sigma(S) + |\omega(S)| + \gamma. \end{aligned}$$

B) $S \in \mathfrak{S}_2$ with $\vartheta_{\frac{1}{2}} = \frac{1}{2}(\sigma(V) - W) \leq \sigma(S) \leq \sigma(V \setminus S) \leq \frac{1}{2}(\sigma(V) + W)$, where the constraints generated by S and $V \setminus S$ are

$$\begin{aligned} \vartheta|\delta(S)| &\geq \sigma(V \setminus S) - \omega(V \setminus S) + \gamma, \\ \vartheta|\delta(S)| &\geq \sigma(S) - \omega(S) + \gamma, \end{aligned}$$

which is equivalent to

$$\begin{aligned} \vartheta|\delta(S)| &\geq \max\{\sigma(V \setminus S) - \omega(V \setminus S), \sigma(S) - \omega(S)\} + \gamma \\ &= \frac{1}{2}(\sigma(V \setminus S) - \omega(V \setminus S) + \sigma(S) - \omega(S) \\ &\quad + |\sigma(V \setminus S) - \omega(V \setminus S) - \sigma(S) + \omega(S)|) + \gamma \\ &= \frac{1}{2}(\sigma(V) - W) + \frac{1}{2}|\sigma(V) - W - 2\sigma(S) + 2\omega(S)| + \gamma \\ &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| + \gamma. \end{aligned}$$

Therefore, we have

$$\begin{aligned} \Theta^\gamma = \{(\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta|\delta(S)| &\geq \sigma(S) + |\omega(S)| + \gamma & \forall S \in \mathfrak{S}_1, \\ \vartheta|\delta(S)| &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| + \gamma & \forall S \in \mathfrak{S}_2, \\ \omega(V) &= W \}, \end{aligned}$$

which is equivalent to the stated conjunction. \square

Remarks: We observe the following symmetry for all $\emptyset \neq S \subsetneq V$ due to $\omega(V) = W$:

$$\begin{aligned} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(V \setminus S) + \omega(V \setminus S)| &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(V) + \sigma(S) + W - \omega(S)| \\ &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - 2\vartheta_{\frac{1}{2}} + \sigma(S) - \omega(S)| \\ &= \vartheta_{\frac{1}{2}} + |-\vartheta_{\frac{1}{2}} + \sigma(S) - \omega(S)| \\ &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)|. \end{aligned} \tag{4.4}$$

In particular, for a vertex set S with $\sigma(S) = \frac{1}{2}\sigma(V)$, where we have $S, V \setminus S \in \mathfrak{S}_2$, the inequalities for S and its complement are thus equivalent.

Furthermore, by resolving the absolute value, we can also formulate

$$\begin{aligned} \Theta^\gamma = \{(\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta|\delta(S)| &\geq \sigma(S) + \omega(S) + \gamma & \forall S \in \mathfrak{S}_1, \\ \vartheta|\delta(S)| &\geq 2\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S) + \gamma & \forall S \in \mathfrak{S}_2, \\ \vartheta|\delta(S)| &\geq \sigma(S) - \omega(S) + \gamma & \forall S \in \mathfrak{S}_1 \cup \mathfrak{S}_2, \\ \omega(V) &= W \} \end{aligned}$$

or

$$\Theta^\gamma = \left\{ (\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \begin{aligned} \sigma(S) - \vartheta|\delta(S)| + \gamma \leq \omega(S) \leq \vartheta|\delta(S)| - \gamma - \sigma(S) & \quad \forall S \in \mathfrak{S}_1, \\ \sigma(S) - \vartheta|\delta(S)| + \gamma \leq \omega(S) \leq \vartheta|\delta(S)| - \gamma + \sigma(S) - 2\vartheta_{\frac{1}{2}} & \quad \forall S \in \mathfrak{S}_2, \\ \omega(V) = W \end{aligned} \right\}.$$

As it can be seen in the latter formulation, we have halved the number of vertex sets but now have two inequalities for each of them. Furthermore, the choice of ϑ mainly influences the available ω -values. Once we have found the optimal value of our problems, the value of ϑ , we also need to know a corresponding weighting ω . Throughout the following work, we thus refer to the following set several times:

Definition 4.13. For $\vartheta \in \mathbb{R}$ let

$$\begin{aligned} \Omega^\gamma(\vartheta) &:= \{\omega : (\vartheta, \omega) \in \Theta^\gamma\} \\ &= \left\{ \omega \in \mathbb{R}^V : \begin{aligned} \sigma(S) - \vartheta|\delta(S)| + \gamma \leq \omega(S) \leq \vartheta|\delta(S)| - \gamma - \sigma(S) & \quad \forall S \in \mathfrak{S}_1, \\ \sigma(S) - \vartheta|\delta(S)| + \gamma \leq \omega(S) \leq \vartheta|\delta(S)| - \gamma + \sigma(S) - 2\vartheta_{\frac{1}{2}} & \quad \forall S \in \mathfrak{S}_2, \\ \omega(V) = W \end{aligned} \right\}. \end{aligned}$$

Clearly, $\Omega^\gamma(\vartheta)$ is empty for $\vartheta < \vartheta_W^\gamma$ with ϑ_W^γ being the optimal value of the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM. In turn, if we have $\omega^* \in \Omega^\gamma(\vartheta_W^\gamma)$, we have

$$(\vartheta_W^\gamma, \omega^*) \in \arg \min \{(1, \mathbb{O})^\top \lambda : \lambda \in \Theta\},$$

which means we have found an optimal solution to the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM.

With the formulation of the polyhedron as before, we can also very easily derive a trivial lower bound on the objective value of the strength-only problem, and thus also on the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM, by

Corollary 4.14. *We have*

$$\vartheta_W^\gamma \geq \max \left\{ \max_{S \in \mathfrak{S}_1} \frac{\sigma(S) + \gamma}{|\delta(S)|}, \max_{S \in \mathfrak{S}_2} \frac{\vartheta_{\frac{1}{2}} + \gamma}{|\delta(S)|} \right\}.$$

Proof. Clear due to the absolute values in the formulation of $\Theta^\gamma(S)$ in Definition 4.11. \square

Remark: These bounds ensure that, for each cut, the lower bound on the cut weighting is actually lower than the upper bound, hence the intervals as formulated in Definition 4.13 for $\Omega^\gamma(\vartheta)$ are non-empty. Note that this does not imply that $\Omega^\gamma(\vartheta)$ is non-empty, as the different cuts might interfere with each other.

4.2.4 Case Distinction

As we need to deal with the different problems over several case distinctions depending on the specific input parameters, we have illustrated the partitioning of the instances in Figure 4.2. Our deductions in the following work follow the path of a depth-first search from left to right through the tree. Although some of the terms and notations are introduced later, this serves as a guideline through the text.

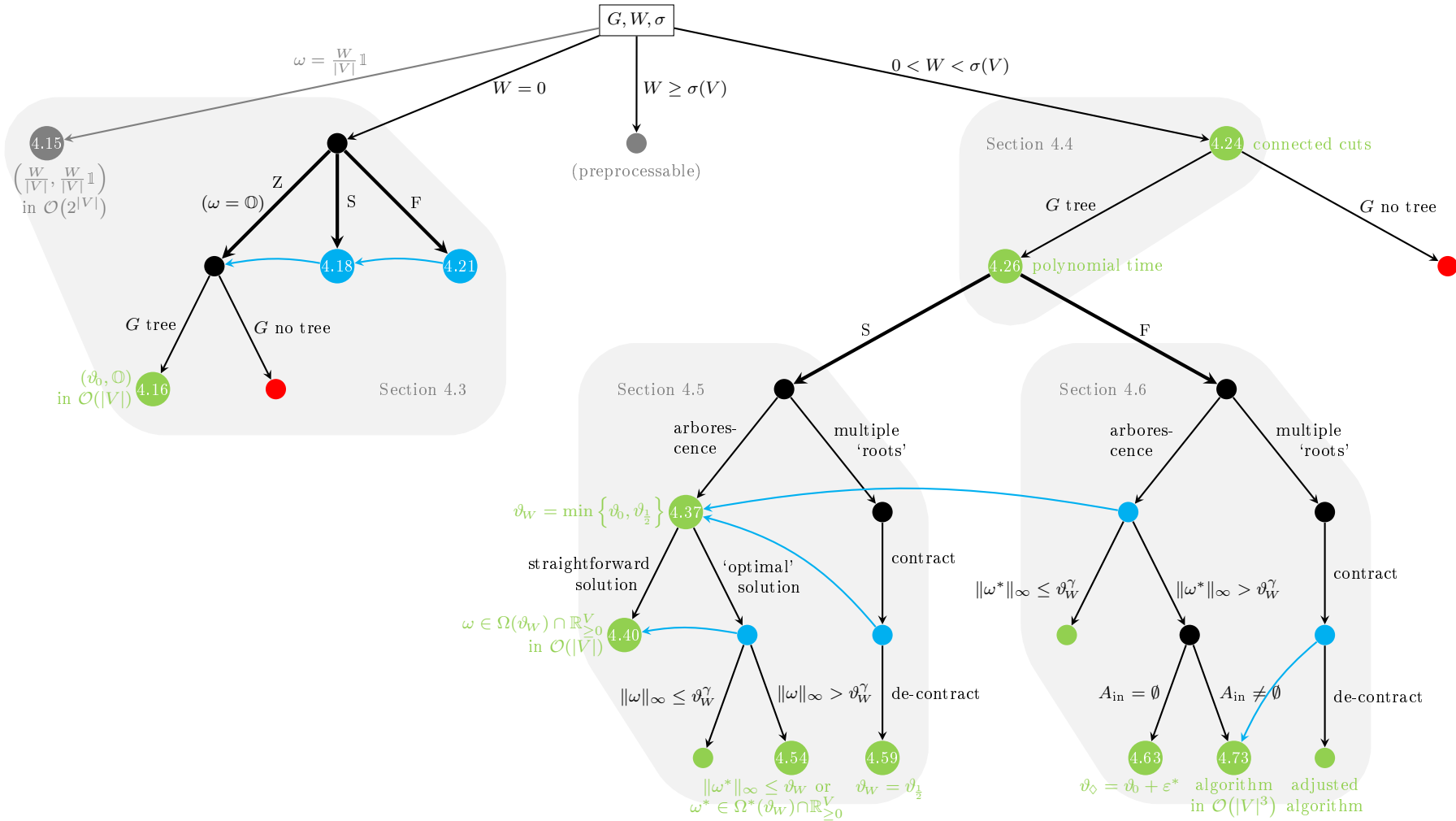


Figure 4.2: Illustration of the case distinctions, where the labels Z, S and F stand for ZERO, STRENGTH-ONLY and the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM, respectively. The nodes mark properties that are fulfilled in the whole subtree and black arcs indicate the branching instructions. The blue nodes with curved arcs indicate where we refer to the results of other branches, on which further branching is based on. Green nodes mark resolved cases, red nodes open ones

4.3 Special Cases

A first option to approach the formulated problems is to evaluate certain specific cases. Two of them are presented in this section. We start with a presumably trivial step of equally distributing the total weight over all vertices. As the next step, we consider a total weight of 0 and show the validity of setting $\omega = \mathbb{0}$ in this case. In contrast to the trivial weighting, we can obtain a first complexity result for this simplified problem version.

4.3.1 Trivial Weighting

By examining the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM, a first supposedly simple step and the best we can do in terms of the weights is equally distributing the total weight with

$$\omega^\ominus := \frac{W}{|V|}\mathbb{1}.$$

With this we have

$$\|\omega^\ominus\|_\infty = \frac{W}{|V|},$$

which is a bound that is exceeded for all other choices of ω . By replacing $\omega(S)$ with $\frac{W}{|V|}|S|$ accordingly in the cut inequalities for all $\emptyset \neq S \subsetneq V$, they reduce to

$$\begin{aligned} \vartheta|\delta(S)| &\geq \min \left\{ \sigma(S) + \frac{W}{|V|}|S|, \sigma(V \setminus S) - \frac{W}{|V|}|V \setminus S| \right\} + \gamma \\ &= \min \left\{ \sigma(S) + \frac{W}{|V|}|S|, \sigma(V) - \sigma(S) - \frac{W}{|V|}(|V| - |S|) \right\} + \gamma \\ &= \min \left\{ \sigma(S), \sigma(V) - W - \sigma(S) \right\} + \frac{W}{|V|}|S| + \gamma \\ &= \frac{W}{|V|}|S| + \gamma + \begin{cases} \sigma(S) & \text{if } \sigma(S) < \vartheta_{\frac{1}{2}} = \frac{1}{2}(\sigma(V) - W), \\ 2\vartheta_{\frac{1}{2}} - \sigma(S) & \text{otherwise} \end{cases} \\ &= \frac{W}{|V|}|S| + \vartheta_{\frac{1}{2}} - |\sigma(S) - \vartheta_{\frac{1}{2}}| + \gamma. \end{aligned}$$

Obviously, this does not simplify the problem significantly at first sight.

However, in case we set the weights like this, is it also possible to set ϑ as low as the weights, thus $\vartheta = \frac{W}{|V|}$, too? With this we can deduce for the inequalities

$$\begin{aligned} \frac{W}{|V|}|\delta(S)| &\geq \frac{W}{|V|}|S| + \vartheta_{\frac{1}{2}} - |\sigma(S) - \vartheta_{\frac{1}{2}}| + \gamma \\ \Leftrightarrow \frac{W}{|V|}(|\delta(S)| - |S|) &\geq \vartheta_{\frac{1}{2}} - |\sigma(S) - \vartheta_{\frac{1}{2}}| + \gamma. \end{aligned}$$

In case we have some $\emptyset \neq S \subsetneq V$ with $|\delta(S)| \leq |S|$, we do have a contradiction if we also have

$$\vartheta_{\frac{1}{2}} - |\sigma(S) - \vartheta_{\frac{1}{2}}| + \gamma > 0,$$

which is equivalent to

$$\begin{aligned} \vartheta_{\frac{1}{2}} + \gamma &> \sigma(S) - \vartheta_{\frac{1}{2}}, \\ \vartheta_{\frac{1}{2}} + \gamma &> -\sigma(S) + \vartheta_{\frac{1}{2}}. \end{aligned}$$

While the latter is always fulfilled since we have $\gamma > 0$ and $\sigma \geq \mathbb{0}$, the first is given if we have

$$\sigma(S) < \sigma(V) - W + \gamma.$$

With this we can now formulate the following result:

Lemma 4.15. *If we have*

$$\frac{W}{|V|}(|\delta(S)| - |S|) \geq \vartheta_{\frac{1}{2}} - |\sigma(S) - \vartheta_{\frac{1}{2}}| + \gamma \quad \forall \emptyset \neq S \subsetneq V,$$

then $\frac{W}{|V|}(1, \mathbb{1}_V)$ is an optimal solution of the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM.

Unfortunately, there is an exponential number of inequalities to be checked. Thus, we cannot decide the above problem in polynomial time in a straightforward way and do not gain an advantage compared to the general problem.

4.3.2 Zero Weight

By inverting and combining the cut inequalities of the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM, we obtain

$$\begin{aligned} \frac{1}{\vartheta} &\leq \min \left\{ \frac{|\delta(S)|}{\min\{\sigma(S), \sigma(V \setminus S)\} + \gamma} : \emptyset \neq S \subsetneq V \right\} \\ &= \min \left\{ \frac{|\delta(S)|}{\sigma(S) + \gamma} : \emptyset \neq S \subsetneq V, \sigma(S) \leq \sigma(V \setminus S) \right\}. \end{aligned}$$

In this formulation, we see a relation to a graph property called the *edge expansion* of the graph [25]. In our case, we additionally have a weighting σ on the vertices, rather than just counting the number of vertices. This is in turn closely related to the *minimum cut density* as introduced in [40]. There the authors show that, in case of trees, the minimum cut density can be solved by just evaluating those partitions which are derived from cutting at each edge. Thus, the problem can be solved in a time quadratically in the number of vertices if the examined graph is a tree.

However, our problem formulation is slightly different to both mentioned ones. Nevertheless, we can establish a similar result for the most simplified version of our problem in this section. Furthermore, we show that this version is still relevant when dealing with the strength-only or full problem.

Optimal Solution for Trees

By defining

$$\vartheta_0^\gamma(S) := \frac{\min\{\sigma(S), \sigma(V \setminus S)\} + \gamma}{|\delta(S)|},$$

we can formulate

$$\vartheta_0^\gamma = \max \{ \vartheta_0^\gamma(S) : \emptyset \neq S \subsetneq V \}.$$

It is easy to see that we have $\vartheta_0^\gamma(S) = \vartheta_0^\gamma(V \setminus S)$ and $\vartheta_0 \leq \frac{1}{2}\sigma(V) + \gamma$. In case of G being a tree, we investigate fundamental cuts with

$$\vartheta_0^\gamma(e) := \vartheta_0^\gamma(S_e) = \min\{\sigma(S_e), \sigma(V \setminus S_e)\} + \gamma,$$

where $S_e, V \setminus S_e$ are the vertex partitions of the tree derived by cutting at edge e . We can indeed show that we can reduce to these cuts:

Theorem 4.16. *If G is a tree, we have*

$$\vartheta_0^\gamma = \max \{ \vartheta_0^\gamma(e) : e \in E \}.$$

Proof. We proof the claim by contradiction: Assume ϑ_0^γ is maximized by a vertex set S^* and its complement $V \setminus S^*$ with $|\delta(S^*)| = |\delta(V \setminus S^*)| = k > 1$. By shrinking the connected components of the vertex sets S^* and $V \setminus S^*$, we obtain a bipartite tree with $k + 1$ nodes. Since we have $k > 1$, the number of connected components in one of the sets is larger than 1, w.l.o.g. let $V \setminus S^*$ be this set. We split $V \setminus S^* = X \cup Y$ with $X, Y \neq \emptyset$, w.l.o.g. $\sigma(X) \geq \sigma(Y)$, and

$$\delta(V \setminus S^*) = \delta(S^*) = \delta(X) \cup \delta(Y).$$

Furthermore, let $k_X := |\delta(X)| \geq 1$ and $k_Y := |\delta(Y)| \geq 1$. Then we have $k = k_X + k_Y$ and $k_X, k_Y < k$.

We distinguish the following two cases:

a) $\sigma(S^*) + \sigma(Y) \leq \sigma(X)$: Remembering $\sigma \geq \mathbb{O}$, we further have

$$\sigma(S^*) \leq \sigma(X) - \sigma(Y) \leq \sigma(X) + \sigma(Y) = \sigma(V \setminus S^*),$$

and thus

$$\vartheta_0^\gamma(X) = \frac{\sigma(S^*) + \sigma(Y) + \gamma}{k_X} \geq \frac{\sigma(S^*) + \gamma}{k_X} > \frac{\sigma(S^*) + \gamma}{k} = \vartheta_0^\gamma(S^*),$$

which is a contradiction to S^* being a maximizing cut.

b) $\sigma(S^*) + \sigma(Y) > \sigma(X)$: With additionally

$$\sigma(X) + \sigma(S^*) \geq \sigma(X) \geq \sigma(Y),$$

we have

$$\begin{aligned} \vartheta_0^\gamma(X) &= \frac{\sigma(X) + \gamma}{k_X}, \\ \vartheta_0^\gamma(Y) &= \frac{\sigma(Y) + \gamma}{k_Y}. \end{aligned}$$

For S^* , we can deduce

$$\vartheta_0^\gamma(S^*) = \frac{\min\{\sigma(S^*), \sigma(X) + \sigma(Y)\} + \gamma}{k} \leq \frac{\sigma(X) + \sigma(Y) + \gamma}{k_X + k_Y}.$$

Assuming $\vartheta_0^\gamma(S^*) \geq \vartheta_0^\gamma(Y)$, we get

$$\begin{aligned} &\frac{\sigma(X) + \sigma(Y) + \gamma}{k_X + k_Y} \geq \frac{\sigma(Y) + \gamma}{k_Y} \\ \Rightarrow &k_Y \sigma(X) + k_Y \sigma(Y) + k_Y \gamma \geq k_X \sigma(Y) + k_Y \sigma(Y) + k_X \gamma + k_Y \gamma \\ \Rightarrow &k_Y \sigma(X) \geq k_X \sigma(Y) + k_X \gamma \\ \Rightarrow &k_Y \sigma(X) + k_X \sigma(X) + k_Y \gamma + k_X \gamma > k_X \sigma(Y) + k_X \sigma(X) + k_X \gamma \\ \Rightarrow &\frac{\sigma(X) + \gamma}{k_X} > \frac{\sigma(X) + \sigma(Y) + \gamma}{k_X + k_Y} \\ \Rightarrow &\vartheta_0^\gamma(X) > \vartheta_0^\gamma(S^*). \end{aligned}$$

This means we have either $\vartheta_0^\gamma(X) > \vartheta_0^\gamma(S^*)$ or $\vartheta_0^\gamma(Y) > \vartheta_0^\gamma(S^*)$, which is a contradiction to S^* being a maximizing cut again. \square

With this we can easily deduce

Corollary 4.17. *If G is a tree, we can find an optimal solution for the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|E|) = \mathcal{O}(|V|)$.*

Although $\gamma = 0$ is not allowed by the problem restrictions derived in Section 4.2, we can reuse our notation to refer to the ‘gapless’ problem as a part of the gapped problem. However, we drop the superscript γ in this case and define

$$\vartheta_0 := \max_{\emptyset \neq S \subsetneq V} \frac{\min\{\sigma(S), \sigma(V \setminus S)\}}{|\delta(S)|}.$$

It is easy to see that, for G being a tree, we have

$$\vartheta_0^\gamma = \vartheta_0 + \gamma.$$

We refer to the problem of finding ϑ_0 as the ZERO WEIGHT DISTRIBUTION PROBLEM (without ‘gapped’) in the following.

Relation to Other Problems

As we have seen, the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM is easy to solve for trees. But does this provide any benefit for the original problems that we want to solve in case of $W = 0$? By considering the ω ’s again, we see that they are subtracted on the right-hand side of the inequalities in some cases. Thus, although they sum up to 0, it might be possible that they allow for a smaller strength ϑ than achieved by omitting the ω ’s. However, with the following result, we see this is not the case:

Theorem 4.18. *For $W = 0$, there is an optimal solution to the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM with $\omega = \mathbb{0}$.*

Proof. With

$$\begin{aligned} \omega(S) + \omega(V \setminus S) &= W = 0 \\ \Leftrightarrow \omega(S) &= -\omega(V \setminus S) \end{aligned}$$

for $\emptyset \neq S \subsetneq V$, we get

$$\begin{aligned} \vartheta_W^\gamma &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max_{\emptyset \neq S \subsetneq V} \left\{ \frac{\min\{\sigma(S) + \omega(S), \sigma(V \setminus S) - \omega(V \setminus S)\} + \gamma}{|\delta(S)|} \right\} \\ &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max_{\emptyset \neq S \subsetneq V} \left\{ \frac{\min\{\sigma(S) + \omega(S), \sigma(V \setminus S) + \omega(S)\} + \gamma}{|\delta(S)|} \right\} \\ &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max \left\{ \begin{array}{l} \max_{\substack{\emptyset \neq S \subsetneq V \\ \sigma(S) \leq \sigma(V \setminus S)}} \frac{\sigma(S) + \omega(S) + \gamma}{|\delta(S)|}, \quad \max_{\substack{\emptyset \neq S \subsetneq V \\ \sigma(S) \geq \sigma(V \setminus S)}} \frac{\sigma(V \setminus S) + \omega(S) + \gamma}{|\delta(S)|} \end{array} \right\} \\ &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max \left\{ \begin{array}{l} \max_{\substack{\emptyset \neq S \subsetneq V \\ \sigma(S) \leq \sigma(V \setminus S)}} \frac{\sigma(S) + \omega(S) + \gamma}{|\delta(S)|}, \quad \max_{\substack{\emptyset \neq S \subsetneq V \\ \sigma(S) \leq \sigma(V \setminus S)}} \frac{\sigma(S) + \omega(V \setminus S) + \gamma}{|\delta(V \setminus S)|} \end{array} \right\} \\ &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max_{\emptyset \neq S \subsetneq V} \frac{\sigma(S) + \max\{\omega(S), \omega(V \setminus S)\} + \gamma}{|\delta(S)|} \\ &= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max_{\emptyset \neq S \subsetneq V} \frac{\sigma(S) + \max\{\omega(S), -\omega(S)\} + \gamma}{|\delta(S)|} \end{aligned}$$

$$\begin{aligned}
&= \min_{\substack{\omega \in \mathbb{R}^V \\ \omega(V)=0}} \max_{\substack{\emptyset \neq S \subseteq V \\ \sigma(S) \leq \sigma(V \setminus S)}} \frac{\sigma(S) + |\omega(S)| + \gamma}{|\delta(S)|} \\
&= \max_{\substack{\emptyset \neq S \subseteq V \\ \sigma(S) \leq \sigma(V \setminus S)}} \frac{\sigma(S) + \gamma}{|\delta(S)|} \\
&= \max_{\emptyset \neq S \subseteq V} \frac{\min\{\sigma(S), \sigma(V \setminus S)\} + \gamma}{|\delta(S)|} = \vartheta_0^\gamma,
\end{aligned}$$

where the second but last equation holds due to $\omega = \mathbb{0}$ being feasible. \square

This means, in case of $W = 0$, we can indeed restrict ourselves to the simpler problem. For the strength-only version of our weight distribution problem, we easily see:

Corollary 4.19. *For $W = 0$, every optimal solution ϑ_0^γ to the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM yields an optimal solution $(\vartheta_0^\gamma, \mathbb{0})$ to the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM.*

With these results, we also see that our notation is indeed consistent because we have $\vartheta_W^\gamma = \vartheta_0^\gamma$ for $W = 0$. In combination with the former complexity result for the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM for trees, we can also state

Corollary 4.20. *If G is a tree and $W = 0$, we can find an optimal solution for the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|E|) = \mathcal{O}(|V|)$.*

Proof. Clear from Corollaries 4.17 and 4.19. \square

Analogously, we can now step further to the full problem and derive the following equivalent results:

Theorem 4.21. *For $W = 0$, there is an optimal solution to the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM with $\omega = \mathbb{0}$.*

Proof. This follows from Theorem 4.18 and the fact that, if $\vartheta^* \geq \|\omega^*\|_\infty$ for any optimal solution (ϑ^*, ω^*) of the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM, it is also an optimal solution of the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM by Corollary 4.9. \square

Corollary 4.22. *For $W = 0$, every optimal solution ϑ_0^γ to the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM yields an optimal solution $(\vartheta_0^\gamma, \mathbb{0})$ to the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM.*

Corollary 4.23. *If G is a tree, we can find an optimal solution for the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|E|) = \mathcal{O}(|V|)$.*

4.4 Simplified Description of the Θ -Polyhedron

In the previous section, we have shown that the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM can be solved in linear time in case we consider only trees. In the following, we evaluate the Θ^γ -polyhedron for a fixed γ to establish analogous results for the general STRENGTH-ONLY and FULL GAPPED WEIGHT DISTRIBUTION PROBLEM. In particular, we establish ‘gapless’ versions of our problems in case of trees by shifting γ . With those, we continue to work in the following sections.

As we have already shown the relation of the strength-only and the full gapped problem to the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM for $W = 0$, we refer in the following to the case $W > 0$, but the constructions are usually also valid for $W = 0$. On the other side, we also restrict our considerations to $W < \sigma(V)$ to focus on the cases that are not preprocessable. This implies $\vartheta_{\frac{1}{2}} > 0$, which is indeed necessary for the constructions. We further assume \mathbf{G} and $\sigma \in \mathbb{R}_{\geq 0}^V$ to be given and fixed again.

4.4.1 Connected Vertex Sets

To reduce the complexity of the description of Θ^γ from Definition 4.10, we need to reduce the number of inequalities. By the following result, we can indeed show that some inequalities describing Θ^γ are redundant due to the monotonicity of the σ -sums:

Theorem 4.24. *With $\mathfrak{C} := \{S \in \mathfrak{S}_1 \cup \mathfrak{S}_2 : G[S] \text{ connected and } G[V \setminus S] \text{ connected}\}$, we have*

$$\Theta^\gamma = \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S) \cap \Theta_W.$$

Proof. By Lemma 4.12, the left-hand side is contained in the right-hand side. In the following, we show the reverse direction by establishing the redundancy of the constraints associated with sets in $\mathfrak{S} := \mathfrak{S}_1 \cup \mathfrak{S}_2$ that are not included in \mathfrak{C} . For this we introduce the type $t(S)$ of a vertex set $\emptyset \neq S \subsetneq V$ denoting the number of connected components of the corresponding induced subgraph $G[S]$. Note that every non-empty vertex set has at least a type of 1, and we only have $t(S) = 1$ if $G[S]$ is connected.

As the equality $\omega(V) = W$ is important for the following derivations, we work on the hyperplane Θ_W but do not explicitly mention it for simplicity. Furthermore, we use the following relations:

$$\Delta \quad |a| + |b| \geq |a \pm b|,$$

$$\blacklozenge \quad x = |\pm x| \text{ for } x \geq 0,$$

① for $S \in \mathfrak{S}_1$ we have

$$\begin{aligned} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| &= \underbrace{\vartheta_{\frac{1}{2}} - \sigma(S) + \sigma(S)}_{>0} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| \\ &\blacklozenge = \sigma(S) + |\sigma(S) - \vartheta_{\frac{1}{2}}| + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)| \\ &\Delta \geq \sigma(S) + |\omega(S)|, \end{aligned}$$

② for $S \in \mathfrak{S}_2$ we have

$$\begin{aligned}\sigma(S) + |\omega(S)| &= \vartheta_{\frac{1}{2}} - \underbrace{\vartheta_{\frac{1}{2}} + \sigma(S)}_{\geq 0} + |\omega(S)| \\ &\stackrel{\blacklozenge}{=} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S)| + |\omega(S)| \\ &\stackrel{\blacktriangle}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S) + \omega(S)|.\end{aligned}$$

The first step is to show the sufficiency of the restriction to vertex sets that induce connected subgraphs, which means

$$\bigcap_{S \in \mathfrak{S}} \Theta^\gamma(S) \supseteq \bigcap_{S \in \mathfrak{C}'} \Theta^\gamma(S)$$

with $\mathfrak{C}' = \{S \in \mathfrak{S} : G[S] \text{ connected}\} = \{S \in \mathfrak{S} : t(S) = 1\}$. Suppose this does not hold. Let $S^* \in \mathfrak{S} \setminus \mathfrak{C}' = \{S \in \mathfrak{S} : t(S) > 1\}$ be a vertex set with

$$\Theta^\gamma(S^*) \not\supseteq \bigcap_{S \in \mathfrak{C}'} \Theta^\gamma(S) \quad (\text{i})$$

having minimal type. Then there exist two non-empty vertex sets S_1 and S_2 with $S^* = S_1 \cup S_2$ and $\delta(S_1, S_2) = \delta(S_1) \cap \delta(S_2) = \emptyset$. We have $S_1, S_2 \in \mathfrak{S}$ because of $\sigma(S_1), \sigma(S_2) \leq \sigma(S^*) \leq \frac{1}{2}\sigma(V)$. Due to $t(S_1) + t(S_2) = t(S^*)$ and the minimality of S^* , we have $t(S_1) = t(S_2) = 1$ and therefore $S_1, S_2 \in \mathfrak{C}'$ with

$$\Theta^\gamma(S_1) \cap \Theta^\gamma(S_2) \supseteq \bigcap_{S \in \mathfrak{C}'} \Theta^\gamma(S). \quad (\text{ii})$$

In the following, we show that the inequality defining $\Theta^\gamma(S^*)$ can be derived from the inequalities defining $\Theta^\gamma(S_1)$ and $\Theta^\gamma(S_2)$ by summation. Because

$$\{x \in X : f(x) + g(x) \leq 0\} \supseteq \{x \in X : f(x) \leq 0, g(x) \leq 0\}$$

holds for arbitrary domains X and functions $f, g : X \rightarrow \mathbb{R}$, this results in

$$\Theta^\gamma(S^*) \supseteq \Theta^\gamma(S_1) \cap \Theta^\gamma(S_2), \quad (\text{iii})$$

which is, together with (ii), a contradiction to (i). We have two different cases concerning S^* :

A) $S^* \in \mathfrak{S}_1$: Due to $\sigma(S_1), \sigma(S_2) \leq \sigma(S^*) \leq \vartheta_{\frac{1}{2}}$, we have also $S_1, S_2 \in \mathfrak{S}_1$. From the inequalities defining $\Theta^\gamma(S_i)$,

$$\vartheta|\delta(S_i)| \geq \sigma(S_i) + |\omega(S_i)| + \gamma,$$

for $i = 1, 2$, and

$$|\delta(S^*)| = |\delta(V \setminus S^*)| = |\delta(S_1)| + |\delta(S_2)| - 2|\delta(S_1, S_2)| = |\delta(S_1)| + |\delta(S_2)|,$$

we get

$$\begin{aligned}\vartheta|\delta(S^*)| &= \vartheta|\delta(S_1)| + \vartheta|\delta(S_2)| \\ &\geq \sigma(S_1) + |\omega(S_1)| + \sigma(S_2) + |\omega(S_2)| + 2\gamma \\ &\stackrel{\blacktriangle}{\geq} \sigma(S^*) + |\omega(S_1) + \omega(S_2)| + 2\gamma \\ &= \sigma(S^*) + |\omega(S^*)| + 2\gamma \\ &\geq \sigma(S^*) + |\omega(S^*)| + \gamma,\end{aligned}$$

which provides the constraint defining $\Theta^\gamma(S^*)$.

B) $S^* \in \mathfrak{S}_2$: Due to $S_1, S_2 \in \mathfrak{S} = \mathfrak{S}_1 \cup \mathfrak{S}_2$, there are 3 further possibilities, which follow the same construction as in case A):

a) $S_1, S_2 \in \mathfrak{S}_1$: From

$$\vartheta|\delta(S_i)| \geq \sigma(S_i) + |\omega(S_i)| + \gamma,$$

for $i = 1, 2$, we get

$$\begin{aligned} \vartheta|\delta(S^*)| &= \vartheta|\delta(S_1)| + \vartheta|\delta(S_2)| \\ &\geq \sigma(S_1) + |\omega(S_1)| + \sigma(S_2) + |\omega(S_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \sigma(S^*) + |\omega(S^*)| + 2\gamma \\ &\stackrel{\textcircled{2}}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma. \end{aligned}$$

b) $S_1, S_2 \in \mathfrak{S}_2$: From

$$\vartheta|\delta(S_i)| \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S_i) + \omega(S_i)| + \gamma,$$

for $i = 1, 2$, and $\vartheta_{\frac{1}{2}} > 0$ for $W < \sigma(V)$, we get

$$\begin{aligned} \vartheta|\delta(S^*)| &= \vartheta|\delta(S_1)| + \vartheta|\delta(S_2)| \\ &\geq 2\vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S_1) + \omega(S_1)| + |\vartheta_{\frac{1}{2}} - \sigma(S_2) + \omega(S_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} 2\vartheta_{\frac{1}{2}} + |2\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + 2\gamma \\ &\blacklozenge = \vartheta_{\frac{1}{2}} + |-\vartheta_{\frac{1}{2}}| + |2\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma. \end{aligned}$$

c) $S_1 \in \mathfrak{S}_1, S_2 \in \mathfrak{S}_2$: From

$$\begin{aligned} \vartheta|\delta(S_1)| &\geq \sigma(S_1) + |\omega(S_1)| + \gamma, \\ \vartheta|\delta(S_2)| &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S_2) + \omega(S_2)| + \gamma, \end{aligned}$$

we get

$$\begin{aligned} \vartheta|\delta(S^*)| &= \vartheta|\delta(S_1)| + \vartheta|\delta(S_2)| \\ &\geq \sigma(S_1) + |\omega(S_1)| + \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S_2) + \omega(S_2)| + 2\gamma \\ &\blacklozenge = \vartheta_{\frac{1}{2}} + |-\sigma(S_1) + |\omega(S_1)|| + |\vartheta_{\frac{1}{2}} - \sigma(S_2) + \omega(S_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma. \end{aligned}$$

Therefore, the constraint associated with S^* is redundant if S^* does not induce a connected subgraph. Next we show that the vertex sets whose complement also does not induce a connected subgraph are unnecessary, too, hence

$$\bigcap_{S \in \mathfrak{C}'} \Theta^\gamma(S) \supseteq \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S).$$

Note that, with the former definition of \mathfrak{C}' , we have $\mathfrak{C} = \{S \in \mathfrak{C}' : t(V \setminus S) = 1\}$. We derive an analogous contradiction as before and therefore assume the above relation does not hold. Let $S^* \in \mathfrak{C}' \setminus \mathfrak{C} = \{S \in \mathfrak{S} : t(S) = 1, t(V \setminus S) > 1\}$ be a vertex set with

$$\Theta^\gamma(S^*) \not\supseteq \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S),$$

whose complement $V \setminus S^*$ has minimal type. On the contrary to the first part, we cannot derive straightforwardly that $G[V \setminus S^*]$ consists of two connected components induced by vertex sets in \mathfrak{C} because of $\sigma(V \setminus S^*) \geq \frac{1}{2}\sigma(V)$. Still, we analogously derive the following relations

$$\Theta^\gamma(S^*) \supseteq \Theta^\gamma(Y_1) \cap \Theta^\gamma(Y_2) \supseteq \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S) \quad (\text{iv})$$

for two specific vertex sets Y_1 and Y_2 to be identified first.

With $t(V \setminus S^*) = k$, let

$$V \setminus S^* = \bigcup_{i=1}^k X_i$$

with $t(X_i) = 1$ for all $i = 1, \dots, k$ be a partition of $V \setminus S^*$ into the vertex sets inducing the connected components of $G[V \setminus S^*]$. This is illustrated in Figure 4.3. Since the graph G is connected and we have $t(S^*) = 1$, $G[S^*]$ is connected to each $G[X_i]$ and we also have $t(S^* \cup X_i) = 1$ for all $i = 1, \dots, k$. This can be extended further to any subset of $\{X_i\}_{i=1, \dots, k}$. In particular, we have

$$t(S^* \cup X_1 \cup \dots \cup X_{i-1} \cup X_{i+1} \cup \dots \cup X_k) = t(V \setminus X_i) = 1.$$

We consider two different cases:

- A) Assume there exists $\ell \in \{1, \dots, k\}$ with $\sigma(S^* \cup X_\ell) < \frac{1}{2}\sigma(V)$. Then we have $S^* \cup X_\ell \in \mathfrak{S}$ and therefore $S^* \cup X_\ell \in \mathfrak{C}'$. With $t(V \setminus (S^* \cup X_\ell)) = k - 1 < t(V \setminus S^*)$, this contradicts the choice of S^* for $k > 2$.

In case of $k = 2$, let w.l.o.g. $\ell = 1$ and $\sigma(S^* \cup X_1) < \frac{1}{2}\sigma(V)$. With $S^* \cup X_1 = V \setminus X_2$, we have $V \setminus X_2 \in \mathfrak{S}$ and with $t(V \setminus X_2) = t(X_2) = 1$ therefore $V \setminus X_2 \in \mathfrak{C}$. Furthermore, $\sigma(X_1) < \frac{1}{2}\sigma(V)$ results in $X_1 \in \mathfrak{C}$ and we have

$$\Theta^\gamma(X_1) \cap \Theta^\gamma(V \setminus X_2) \supseteq \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S).$$

- B) Assuming $\sigma(S^* \cup X_i) \geq \frac{1}{2}\sigma(V)$ for all $i = 1, \dots, k$, we get

$$\sigma(V \setminus (S^* \cup X_i)) = \sum_{\substack{j=1 \\ j \neq i}}^k \sigma(X_j) \leq \frac{1}{2}\sigma(V) \quad (\text{v})$$

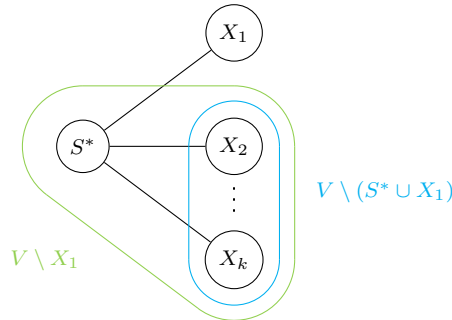


Figure 4.3: Connected components for $S^* \in \mathfrak{C}' \setminus \mathfrak{C}$

and $\sigma(X_i) \leq \frac{1}{2}\sigma(V)$. Therefore, we have $X_i \in \mathfrak{C}$ for all $i = 1, \dots, k$ and

$$\bigcap_{i=1}^k \Theta^\gamma(X_i) \supseteq \bigcap_{S \in \mathfrak{C}} \Theta^\gamma(S).$$

For $k \geq 3$, we can show

$$\Theta^\gamma(Y_{k-1}) \cap \Theta^\gamma(X_k) \supseteq \bigcap_{i=1}^k \Theta^\gamma(X_i),$$

where $Y_j := X_1 \cup \dots \cup X_j$ for $j = 1, \dots, k-1$, by inductively deducing

$$\Theta^\gamma(Y_j) \supseteq \bigcap_{i=1}^j \Theta^\gamma(X_i)$$

with $\Theta^\gamma(Y_1) = \Theta^\gamma(X_1) \supseteq \Theta^\gamma(X_1)$ and the induction step

$$\Theta^\gamma(Y_j) \supseteq \Theta^\gamma(Y_{j-1}) \cap \Theta^\gamma(X_j).$$

The latter follows from similar arguments given to establish (iii) because, due to (v), we have $Y_j \in \mathfrak{S}$ for all $j = 1, \dots, k-1$.

In summary, we have shown that we can restrict the number of sets to $k = 2$ with either

- A) $X_1, V \setminus X_2 \in \mathfrak{S}$ or
- B) $X_1, X_2 \in \mathfrak{S}$ (respectively by renaming Y_{k-1} and X_k).

Because of $\mathfrak{S} = \mathfrak{S}_1 \cup \mathfrak{S}_2$, we use another case distinction in order to establish the first relation of (iv), and thus obtain the desired contradiction. The construction follows the same structure as in the case distinction of the first part. Remember, for $k = 2$, we have

$$\begin{aligned} W &= \omega(S^*) + \omega(X_1) + \omega(X_2), \\ \sigma(V) &= \sigma(S^*) + \sigma(X_1) + \sigma(X_2), \\ |\delta(S^*)| &= |\delta(X_1)| + |\delta(X_2)|. \end{aligned}$$

A) a) $X_1, V \setminus X_2 \in \mathfrak{S}_1$: From the inequalities defining $\Theta^\gamma(X_1)$ and $\Theta^\gamma(V \setminus X_2)$,

$$\begin{aligned} \vartheta|\delta(X_1)| &\geq \sigma(X_1) + |\omega(X_1)| + \gamma, \\ \vartheta|\delta(X_2)| &\geq \sigma(V \setminus X_2) + |\omega(V \setminus X_2)| + \gamma, \end{aligned}$$

we get

$$\begin{aligned} \vartheta|\delta(S^*)| &= \vartheta|\delta(X_1)| + \vartheta|\delta(X_2)| \\ &\geq \sigma(X_1) + |\omega(X_1)| + \sigma(V \setminus X_2) + |\omega(V \setminus X_2)| + 2\gamma \\ &= \sigma(X_1) + |-\omega(X_1)| + \sigma(S^*) + \sigma(X_1) + |W - \omega(X_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \sigma(S^*) + 2\sigma(X_1) + |W - \omega(X_2) - \omega(X_1)| + 2\gamma \\ &\geq \sigma(S^*) + |\omega(S^*)| + \gamma, \end{aligned}$$

which is the inequality defining $\Theta^\gamma(S^*)$ in case of $S^* \in \mathfrak{S}_1$. For $S^* \in \mathfrak{S}_2$, we can extend the chain of relations by

$$\stackrel{\textcircled{2}}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma,$$

which results in the inequality defining $\Theta^\gamma(S^*)$ in this case.

b) $X_1, V \setminus X_2 \in \mathfrak{S}_2$: Due to the symmetry of (4.4), we have

$$\begin{aligned} \vartheta|\delta(X_2)| &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(V \setminus X_2) + \omega(V \setminus X_2)| + \gamma \\ &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(X_2) + \omega(X_2)| + \gamma. \end{aligned}$$

Together with

$$\vartheta|\delta(X_1)| \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(X_1) + \omega(X_1)| + \gamma,$$

we get analogously with

$$\begin{aligned} \vartheta|\delta(S^*)| &= \vartheta|\delta(X_1)| + \vartheta|\delta(X_2)| \\ &\geq 2\vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(X_1) + \omega(X_1)| + |\vartheta_{\frac{1}{2}} - \sigma(X_2) + \omega(X_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} 2\vartheta_{\frac{1}{2}} + |2\vartheta_{\frac{1}{2}} - \sigma(V) + \sigma(S^*) + W - \omega(S^*)| + 2\gamma \\ &= 2\vartheta_{\frac{1}{2}} + |\sigma(S^*) - \omega(S^*)| + 2\gamma \\ &= \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}}| + |-\sigma(S^*) + \omega(S^*)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma \end{aligned}$$

the inequality defining $\Theta^\gamma(S^*)$ for $S^* \in \mathfrak{S}_2$ or by further extending the chain of relations with

$$\stackrel{\textcircled{1}}{\geq} \sigma(S^*) + |\omega(S^*)| + \gamma$$

the inequality for $S^* \in \mathfrak{S}_1$.

c) $X_1 \in \mathfrak{S}_1, V \setminus X_2 \in \mathfrak{S}_2$: Taking advantage of the symmetry again, we have

$$\begin{aligned} \vartheta|\delta(X_1)| &\geq \sigma(X_1) + |\omega(X_1)| + \gamma, \\ \vartheta|\delta(X_2)| &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(X_2) + \omega(X_2)| + \gamma \end{aligned}$$

and get

$$\begin{aligned} \vartheta|\delta(S^*)| &\geq \sigma(X_1) + |\omega(X_1)| + \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(X_2) + \omega(X_2)| + 2\gamma \\ &\blacklozenge = \vartheta_{\frac{1}{2}} + |\sigma(X_1)| + |-\omega(X_1)| + |-\vartheta_{\frac{1}{2}} + \sigma(X_2) - \omega(X_2)| + 2\gamma \\ &\stackrel{\Delta}{\geq} \vartheta_{\frac{1}{2}} + |-\vartheta_{\frac{1}{2}} + \sigma(V) - W - \sigma(S^*) + \omega(S^*)| + 2\gamma \\ &= \vartheta_{\frac{1}{2}} + |\frac{1}{2}\sigma(V) - \frac{1}{2}W - \sigma(S) + \omega(S)| \\ &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma, \end{aligned}$$

respectively further for $S^* \in \mathfrak{S}_1$

$$\stackrel{\textcircled{1}}{\geq} \sigma(S^*) + |\omega(S^*)| + \gamma.$$

d) $X_1 \in \mathfrak{S}_2, V \setminus X_2 \in \mathfrak{S}_1$: Since

$$\sigma(V \setminus X_2) = \sigma(S^*) + \sigma(X_1) < \vartheta_{\frac{1}{2}}$$

contradicts to $\sigma(X_1) \geq \vartheta_{\frac{1}{2}}$, this is an invalid combination.

B) a) $X_1, X_2 \in \mathfrak{S}_1$: From

$$\vartheta|\delta(X_i)| \geq \sigma(X_i) + |\omega(X_i)| + \gamma,$$

for $i = 1, 2$, we get

$$\begin{aligned}
 \vartheta|\delta(S^*)| &\geq \sigma(X_1) + |\omega(X_1)| + \sigma(X_2) + |\omega(X_2)| + 2\gamma \\
 &\stackrel{\Delta}{\geq} \sigma(V) - \sigma(S^*) + |W - \omega(S^*)| + 2\gamma \\
 &= \frac{1}{2}\sigma(V) - \frac{1}{2}W + \frac{1}{2}W + \underbrace{\frac{1}{2}\sigma(V) - \sigma(S^*)}_{\geq 0} + |W - \omega(S^*)| + 2\gamma \\
 &\blacklozenge = \vartheta_{\frac{1}{2}} + |\frac{1}{2}W + \frac{1}{2}\sigma(V) - \sigma(S^*)| + |-W + \omega(S^*)| + 2\gamma \\
 &\stackrel{\Delta}{\geq} \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(S^*) + \omega(S^*)| + \gamma,
 \end{aligned}$$

respectively, further for $S^* \in \mathfrak{S}_1$

$$\stackrel{\textcircled{1}}{\geq} \sigma(S^*) + |\omega(S^*)| + \gamma.$$

b) $X_1, X_2 \in \mathfrak{S}_2$: This case is analogous to A)b).

c) $X_1 \in \mathfrak{S}_1, X_2 \in \mathfrak{S}_2$: This case is analogous to A)c).

Therefore, the constraint for S^* is also redundant if $V \setminus S^*$ is not connected. \square

Although the Θ^γ -polyhedron can now be described with less inequalities, their number might still be exponential in an arbitrary graph. In the following, we thus concentrate on trees and take advantage of their much simpler structure. Note that this simplification is still useful in practice because, by the requirements of an embedding, we can always restrict ourselves to trees by simply ignoring surplus edges.

4.4.2 Arcs in Trees

In this section, we introduce new notations to deal with the problems on trees. As we only consider the smaller σ -sum of the two vertex sets corresponding to a fundamental cut, we can give the corresponding edge a direction. Thus, we define:

Definition 4.25. For G being a tree, let

$$\mathbf{A} := \{a = (v, w) : \{v, w\} \in E, \sigma(T_a) \leq \frac{1}{2}\sigma(V)\},$$

where T_a denotes the vertex set of the subtree which is derived from cutting the tree G at edge $\{v, w\} \in E$ and contains w . Further let

$$\begin{aligned}
 \mathbf{A}_{\text{in}} &:= \{a \in \mathbf{A} : \sigma(T_a) \geq \vartheta_{\frac{1}{2}}\} = \{a \in \mathbf{A} : T_a \in \mathfrak{S}_2\}, \\
 \mathbf{A}_{\text{out}} &:= \{a \in \mathbf{A} : \sigma(T_a) < \vartheta_{\frac{1}{2}}\} = \{a \in \mathbf{A} : T_a \in \mathfrak{S}_1\}
 \end{aligned}$$

denote the *inner* and *outer arcs*, respectively. We have $\mathbf{A} = \mathbf{A}_{\text{in}} \cup \mathbf{A}_{\text{out}}$.

We use the terms ‘inner’ and ‘outer’ arcs due to their appearance in the tree in relation to the root. An example tree is shown in Figure 4.4.

Using this definition, we can easily deduce a first complexity result for G being restricted to trees.

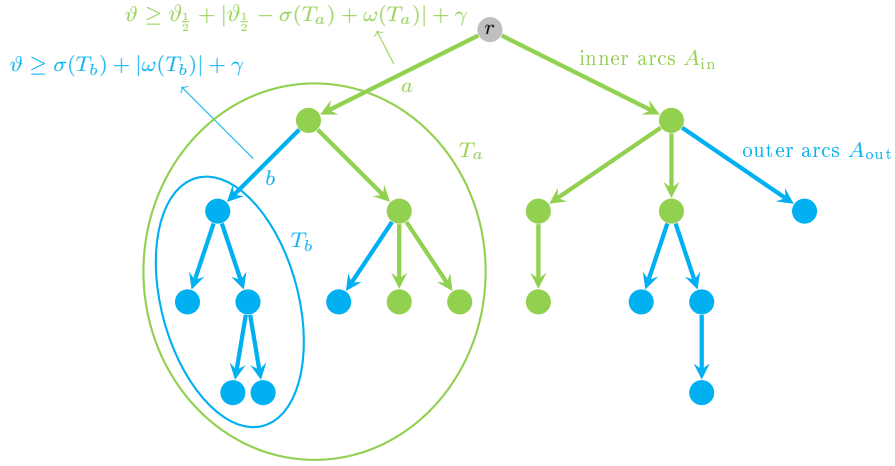


Figure 4.4: Example tree with inner and outer arcs

Theorem 4.26. *If G is a tree and A as defined before, we get*

$$\Theta^\gamma = \bigcap_{a \in A} \Theta^\gamma(T_a) \cap \Theta_W.$$

Therefore, the FULL GAPPED WEIGHT DISTRIBUTION PROBLEM and the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM can be solved in polynomial time.

Proof. With the given graph being a tree, the set \mathfrak{C} from Theorem 4.24 equals $\{T_a : a \in A\}$ and we have $|A| \leq 2(|V| - 1)$ inequalities describing Θ^γ . Thus, the claim follows. \square

In the following, we only consider the case of G being a tree and keep using the notations defined before. Because of $|\delta(T_a)| = 1$ for all $a \in A$ and with the former definitions, the set of feasible solutions Θ^γ can be described for trees G as

$$\Theta^\gamma = \left\{ (\vartheta, \omega) : \begin{aligned} \vartheta &\geq \sigma(T_a) + |\omega(T_a)| + \gamma & \forall a \in A_{\text{out}}, \\ \vartheta &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a)| + \gamma & \forall a \in A_{\text{in}}, \\ \omega(V) &= W \end{aligned} \right\} \quad (4.5)$$

and the STRENGTH-ONLY [FULL] GAPPED WEIGHT DISTRIBUTION PROBLEM can be written as the linear program

$$\begin{array}{ll} \min & \vartheta \\ \text{s.t.} & \vartheta \geq \sigma(T_a) + |\omega(T_a)| + \gamma \quad \forall a \in A_{\text{out}}, \\ & \vartheta \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a)| + \gamma \quad \forall a \in A_{\text{in}}, \\ & \omega(V) = W, \\ & [\vartheta \geq \|\omega\|_\infty,] \\ & \vartheta \in \mathbb{R}, \omega \in \mathbb{R}^V. \end{array}$$

Although the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM is a very specific special case of our problems, we see that its optimal value ϑ_0^γ appears again several times. Analogously to above, using the arc definition, we can also derive a different description of this problem with:

Corollary 4.27. *If G is a tree, the optimal value of the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM is*

$$\begin{aligned}\vartheta_0^\gamma &= \max \{ \sigma(T_a) + \gamma : a \in A \} \\ &= \max \{ \sigma(T_a) : a \in A \} + \gamma \\ &= \vartheta_0 + \gamma.\end{aligned}$$

Proof. For $W = 0$, we have $\vartheta_{\frac{1}{2}} = \frac{1}{2}\sigma(V)$ and therefore $A = A_{\text{out}} \cup \{a \in A_{\text{in}} : \sigma(T_a) = \vartheta_{\frac{1}{2}}\}$. Applying theorems 4.26 and 4.21 results in the claim. \square

4.4.3 Eliminating the Gap Requirement

By the reformulation of the previous section, we can see that γ is indeed only an additive constant for trees. Thus, we simplify the problem further at this stage by substituting ϑ with $\vartheta + \gamma$ and define the ‘gapless’ problems:

STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES. Given a tree $G = (V, E)$, $\sigma \in \mathbb{R}_{\geq 0}^V$, $W \in \mathbb{R}_{\geq 0}$ [and $\gamma \in \mathbb{R}_{> 0}$], find ϑ and ω that solve

$$\begin{aligned}\min & \vartheta \\ \text{s.t.} & \vartheta \in \mathbb{R}, \omega \in \mathbb{R}^V, \\ & \vartheta \geq \sigma(T_a) + |\omega(T_a)| \quad \forall a \in A_{\text{out}}, \\ & \vartheta \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a)| \quad \forall a \in A_{\text{in}}, \\ & \omega(V) = W, \\ & [\vartheta \geq \|\omega\|_\infty - \gamma].\end{aligned}$$

Analogously to the GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM, we reuse the notation from before and drop the superscript γ when considering it to be 0. Thus, let

$$\begin{aligned}\Theta &:= \{ (\vartheta, \omega) : \vartheta \geq \sigma(T_a) + |\omega(T_a)| \quad \forall a \in A_{\text{out}}, \\ & \quad \vartheta \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a)| \quad \forall a \in A_{\text{in}}, \\ & \quad \omega(V) = W \}.\end{aligned}$$

It is easy to see that the sets Θ and Θ^γ are equal apart from the linear transformation on ϑ :

$$\Theta^\gamma = \{ \theta + (\gamma, \mathbb{0}) : \theta \in \Theta \}.$$

Let ϑ_W and ϑ_\diamond be the optimal values of the above problem in the strength-only and the full version, respectively. Note that we already took the summand γ out of the objective because it is only a constant. Therefore, we have

$$\begin{aligned}\vartheta_W^\gamma &= \min \{ \vartheta : (\vartheta, \omega) \in \Theta^\gamma \} \\ &= \min \{ \vartheta + \gamma : (\vartheta + \gamma, \omega) \in \Theta^\gamma \} \\ &= \min \{ \vartheta + \gamma : (\vartheta, \omega) \in \Theta \} \\ &= \min \{ \vartheta : (\vartheta, \omega) \in \Theta \} + \gamma \\ &= \vartheta_W + \gamma.\end{aligned}$$

With

$$\Phi^{-\gamma} := \{(\vartheta, \omega) \in \mathbb{R} \times \mathbb{R}^V : \vartheta \geq \|\omega\|_\infty - \gamma\},$$

we can analogously deduce

$$\Theta^\gamma \cap \Phi = \{\theta + (\gamma, \mathbb{O}) : \theta \in \Theta \cap \Phi^{-\gamma}\}$$

and thus

$$\begin{aligned} \vartheta_\diamond^\gamma &= \min \{ \vartheta : (\vartheta, \omega) \in \Theta^\gamma \cap \Phi \} \\ &= \min \{ \vartheta + \gamma : (\vartheta + \gamma, \omega) \in \Theta^\gamma \cap \Phi \} \\ &= \min \{ \vartheta + \gamma : (\vartheta, \omega) \in \Theta \cap \Phi^{-\gamma} \} \\ &= \min \{ \vartheta : (\vartheta, \omega) \in \Theta \cap \Phi^{-\gamma} \} + \gamma \\ &= \vartheta_\diamond + \gamma. \end{aligned}$$

However, it is important to mention that, in contrast to the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES, where we can indeed eliminate the gap from the problem definition, the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES and thus its optimal value ϑ_\diamond are not completely independent of γ because it still appears in the last inequality. Therefore, it is rather only a shift of the gap requirement. Nevertheless, we use the term *gapless* to refer to the last problem versions for simplicity.

From here on, we only work with the gapless variants of our problems. Once we have found the optimal value of one of those problems, we also want to extract a suitable weighting from it. With the previous results and reformulations, we can also simplify the description of $\Omega^\gamma(\vartheta)$ from Definition 4.13 for a given shifted ϑ in case of trees. We reuse the change of the notation for all remaining objects and drop the superscript γ analogously wherever we set it to 0.

Thus, we can state

Lemma 4.28. *For G being a tree, we have*

$$\begin{aligned} \Omega(\vartheta) = \{ \omega \in \mathbb{R}^V : & \sigma(T_a) - \vartheta \leq \omega(T_a) \leq \vartheta - \sigma(T_a) \quad \forall a \in A_{\text{out}}, \\ & \sigma(T_a) - \vartheta \leq \omega(T_a) \leq \vartheta + \sigma(T_a) - 2\vartheta_{\frac{1}{2}} \quad \forall a \in A_{\text{in}}, \\ & \omega(V) = W \}. \end{aligned}$$

Proof. Resolving the absolute values in (4.5) immediately leads to the formulation claimed in the lemma. \square

From the above description of $\Omega(\vartheta)$, we extract the bounds describing valid weights for a certain ϑ to use them in later deductions.

Definition 4.29. For $\vartheta \geq \vartheta_W$ and $a \in A$, let

$$\begin{aligned} \omega^+(\mathbf{T}_a, \vartheta) &:= \begin{cases} \vartheta - \sigma(T_a) & \text{if } a \in A_{\text{out}}, \\ \vartheta + \sigma(T_a) - 2\vartheta_{\frac{1}{2}} & \text{if } a \in A_{\text{in}}, \end{cases} \\ \omega^-(\mathbf{T}_a, \vartheta) &:= \sigma(T_a) - \vartheta \end{aligned}$$

denote the upper, respectively, lower bound on the weight of the subtree T_a .

Remarks:

- Due to $\sigma(T_a) < \vartheta_{\frac{1}{2}}$ for $a \in A_{\text{out}}$ by definition and $\sigma(T_a) \leq \vartheta_0$ by Corollary 4.27, we have $\sigma(T_a) \leq \vartheta_w$, thus $\omega^-(T_a, \vartheta) \leq 0 \leq \omega^+(T_a, \vartheta)$, for all $a \in A_{\text{out}}$ for $\vartheta \geq \vartheta_w$.
- Due to $\sigma(T_a) < \vartheta_{\frac{1}{2}}$ for $a \in A_{\text{out}}$ and $\sigma(T_a) \geq \vartheta_{\frac{1}{2}}$ for $a \in A_{\text{in}}$, we have $\omega^+(T_a, \vartheta) \geq 0$ for all $a \in A$ for $\vartheta \geq \vartheta_{\frac{1}{2}}$.
- Because of the monotonicity of σ , we have $\omega^-(T_b, \vartheta) \leq \omega^-(T_a, \vartheta)$ for all $b = (v, w) \in A$ with $v, w \in T_a$ for all $a \in A$.

With this definition, we can establish a first relation for different values of ϑ :

Corollary 4.30. *For $\vartheta_1 \leq \vartheta_2$, we have $\Omega(\vartheta_1) \subseteq \Omega(\vartheta_2)$.*

Proof. Since we have

$$\omega^+(T_a, \vartheta_1) \leq \omega^+(T_a, \vartheta_2)$$

and

$$\omega^-(T_a, \vartheta_2) \leq \omega^-(T_a, \vartheta_1)$$

for all $a \in A$, we clearly also have $\omega \in \Omega(\vartheta_2)$ for some $\omega \in \Omega(\vartheta_1)$. \square

From this relation, it is easy to see that the set of feasible weights grows the larger we choose ϑ . This mainly becomes relevant when doing the step from the STRENGTH-ONLY to the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES.

4.4.4 Unique Root

By further evaluating the arc set A , we can see that we need to distinguish between two cases for a given tree. The first is handled in this section: If there is no arc with $\sigma(T_a) = \sigma(V \setminus T_a) = \sigma(T_{\bar{a}}) = \frac{1}{2}\sigma(V)$, with $\bar{a} = (w, v)$ for $a = (v, w)$, the direction of each edge in the tree G is well-defined, pointing towards the subtree whose vertices have the smaller sum of σ -values. Thus, the set A forms an arborescence with $|A| = |E| = |V| - 1$, because of the monotonicity of the σ -sums. Let $\mathbf{r} \in V$ be the unique root. Then we have $r \notin T_a$ for all $a \in A$.

In the following part of this work, we mainly concentrate on trees where A forms an arborescence, as we can exploit the advantages of the recursive structure there. Thus, we define:

Definition 4.31. Let

$$\mathbf{P}(v) := \{p \in V : (p, v) \in A\}$$

be the set of predecessors of v . If we have $|P(v)| = 1$, let \mathbf{p}_v denote the unique predecessor, thus $P(v) = \{\mathbf{p}_v\}$. If \mathbf{p}_v exists, we have $(\mathbf{p}_v, v) \in A$ and define

$$\mathbf{T}_v := T_{(\mathbf{p}_v, v)}.$$

Further let

$$\begin{aligned} \mathbf{V}_{\text{in/out}} &:= \{v \in V : (p, v) \in A_{\text{in/out}}, p \in P(v)\} \\ &= \{v \in V : \sigma(T_{(\mathbf{p}_v, v)}) \geq / < \vartheta_{\frac{1}{2}}, p \in P(v)\}, \\ \mathbf{S}_{\text{in/out}}(v) &:= \{s \in V : (v, s) \in A_{\text{in/out}}\} \end{aligned}$$

define the vertices, respectively, successors of v along inner or outer arcs (respectively both if used without subscript).

Remarks: For the unique root r ,

- $P(r) = \emptyset$ and p_r is not defined,
- we have $P(v) = \{p_v\}$ for all $v \in V \setminus \{r\}$,
- we have $r \notin V_{\text{in}}, r \notin V_{\text{out}}$, thus $V = V_{\text{in}} \cup V_{\text{out}} \cup \{r\}$, and
- we further define $\mathbf{T}_r := V$.

With the former definitions, the set of feasible solutions Θ can be described for trees G with A forming an arborescence as

$$\Theta = \left\{ (\vartheta, \omega) : \begin{aligned} \vartheta &\geq \sigma(T_v) + |\omega(T_v)| && \forall v \in V_{\text{out}}, \\ \vartheta &\geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_v) + \omega(T_v)| && \forall v \in V_{\text{in}}, \\ \omega(V) &= W \end{aligned} \right\}. \quad (4.6)$$

In the following, we use both, the arc or the vertex notation, depending on its suitability. In Figure 4.5, we illustrate the inequalities describing $\Omega(\vartheta)$ in the introduced vertex notation. The STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES is equivalently

$$\begin{aligned} \min \quad & \vartheta \\ \text{s.t.} \quad & \vartheta \geq \sigma(T_v) + |\omega(T_v)| && \forall v \in V_{\text{out}}, \\ & \vartheta \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_v) + \omega(T_v)| && \forall v \in V_{\text{in}}, \\ & \omega(V) = W, \\ & [\vartheta \geq \|\omega\|_{\infty},] \\ & \vartheta \in \mathbb{R}, \omega \in \mathbb{R}^V. \end{aligned}$$

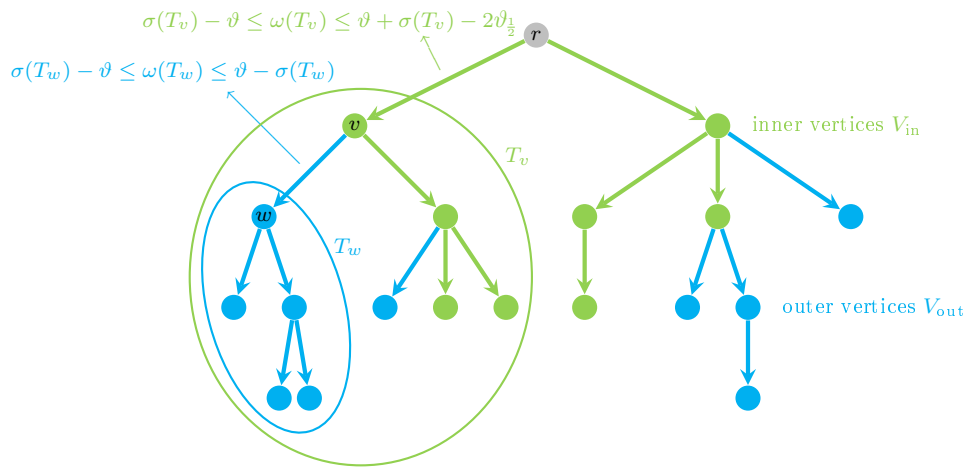


Figure 4.5: Arborescence with inner and outer vertices

4.4.5 Contraction of Multiple Roots

If, in contrast to the assumptions of the previous section, there exists at least one $a \in A$ with $\sigma(T_a) = \sigma(T_{\bar{a}}) = \frac{1}{2}\sigma(V)$, we have the other case concerning the arc set A , where there is not a unique root and A does not form an arborescence as both a and \bar{a} are included in A . With $W > 0$ we have $\frac{1}{2}\sigma(V) > \vartheta_{\frac{1}{2}}$, thus $a, \bar{a} \in A_{\text{in}}$ and therefore always $A_{\text{in}} \neq \emptyset$ in this case. Let

$$A_R := \{a \in A : \sigma(T_a) = \frac{1}{2}\sigma(V)\} \subseteq A_{\text{in}}$$

be the set combining all such arcs. It is easy to see that the edges corresponding to A_R form a path in G over all ‘roots’

$$R := \{v \in V : v \in a, a \in A_R\}.$$

All inner vertices of this path need to have a σ -value equal to 0, so do all vertices in possible subtrees attached to those vertices, otherwise the σ -sums could not be constantly $\frac{1}{2}\sigma(V)$ over all of the subtrees corresponding to the arcs in A_R . Let ℓ and r describe the two endpoints of the path and T_ℓ and T_r be the vertex sets of the two remaining subtrees arising from the removal of the inner vertices of this path. There the arborescence is defined properly. This case and the corresponding notation is illustrated in Figure 4.6. We further observe

- we have $R \subseteq V_{\text{in}}$ and $R \cap V_{\text{out}} = \emptyset$ and thus $V = V_{\text{in}} \cup V_{\text{out}}$,
- we have $|P(v)| = 2$ for all $v \in R \setminus \{\ell, r\}$ and
- $p_\ell, p_r \in R$ exist.

One possibility to deal with these instances as well is to reduce them to the case of unique roots by applying a ‘contraction’:

Definition 4.32. Given an instance (G, σ, W) for the STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES with a tree $G = (V, E)$, where the corresponding arc set A does not have a unique root, $\sigma \in \mathbb{R}_{\geq 0}^V$ and $W \in \mathbb{R}_{\geq 0}$, let \tilde{r} be a new vertex and

$$\begin{aligned} \tilde{V} &:= \{\tilde{r}\} \cup T_\ell \cup T_r \setminus \{\ell, r\}, \\ \tilde{E} &:= \{c(v)c(w) : vw \in E, v, w \in T_\ell \cup T_r, \{v, w\} \neq \{\ell, r\}\}, \end{aligned}$$

for the contraction $c : T_\ell \cup T_r \rightarrow \tilde{V}$ with

$$c(v) = \begin{cases} \tilde{r} & \text{if } v \in \{\ell, r\}, \\ v & \text{otherwise.} \end{cases}$$

From \tilde{E} we can analogously derive the arc set \tilde{A} and have

$$\tilde{A} = \{(c(v), c(w)) : (v, w) \in A, v, w \in T_\ell \cup T_r, \{v, w\} \neq \{\ell, r\}\}.$$

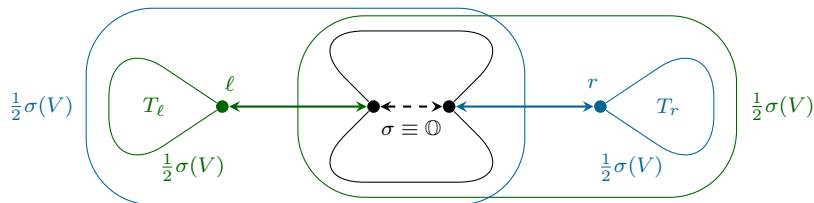


Figure 4.6: Illustration of the σ -distribution for the case of multiple root

Furthermore, let $\tilde{\sigma} \in \mathbb{R}_{\geq 0}^{\tilde{V}}$ with

$$\begin{aligned}\tilde{\sigma}_v &:= \sigma_v & \forall v \in T_\ell \cup T_r \setminus \{\ell, r\}, \\ \tilde{\sigma}_{\tilde{r}} &:= \sigma_\ell + \sigma_r.\end{aligned}$$

With $\tilde{G} := (\tilde{V}, \tilde{E})$ being the resulting tree, we call $(\tilde{G}, \tilde{\sigma}, W)$ the *contracted instance* to the given one for the STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES.

Remarks:

- \tilde{A} forms an arborescence with unique root \tilde{r} .
- The mapping $\tilde{a} := (c(v), c(w)) \in \tilde{A}$ for $a = (v, w) \in A$ with $v, w \in T_\ell$ or $v, w \in T_r$ defines a bijection.
- The set of vertices of the subtrees are preserved with $T_{\tilde{a}} = T_a$ for $\tilde{a} \in \tilde{A}$.
- We have $\tilde{\sigma}(T_{\tilde{a}}) = \sigma(T_a)$ for all $\tilde{a} \in \tilde{A}$.
- We have $\tilde{\sigma}(\tilde{V}) = \sigma(V)$ due to $\sigma_v = 0$ for all $v \in V \setminus (T_\ell \cup T_r)$ and therefore $\tilde{\vartheta}_{\frac{1}{2}} = \vartheta_{\frac{1}{2}}$.
- For $\tilde{a} \in \tilde{A}$ we have $\tilde{a} \in \tilde{A}_{\text{in/out}} \Leftrightarrow a \in A_{\text{in/out}}$.

An example of such a construction is shown in Figure 4.7. The original, uncontracted instance yields a polyhedron of a larger dimension than the contracted one. Thus, we cannot directly derive some kind of boundary condition. However, for some cases we can show, how to extract the original solution from the contracted one.

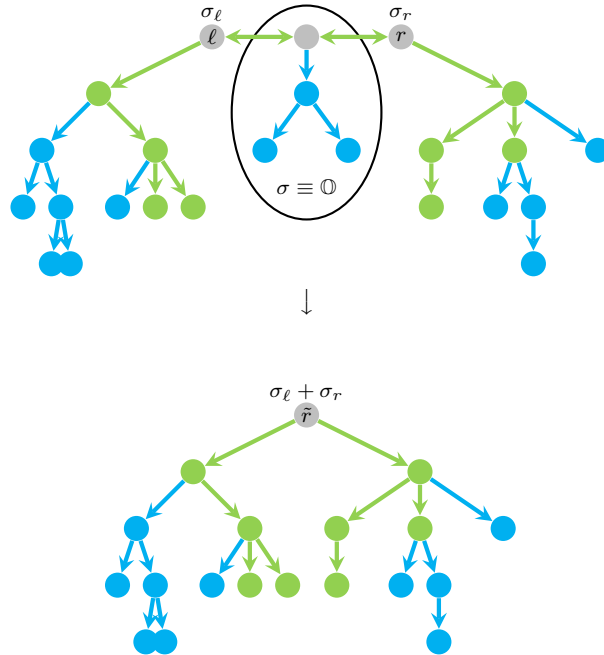


Figure 4.7: Example of a contraction of a tree to obtain a unique root

4.5 Strength-Only Problem on Trees

In this section, we concentrate on the gapless version of our problem, the **STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES**. Thus, we do not need to deal with the gap γ here. Let again the graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, the strengths $\boldsymbol{\sigma} \in \mathbb{R}_{\geq 0}^V$ and the total weight $\mathbf{W} \in \mathbb{R}_{\geq 0}$ be given and fixed. Remember, we assume $\sigma(V) > W > 0$. Thus, we have $0 < \vartheta_{\frac{1}{2}} < \frac{1}{2}\sigma(V)$.

We start our deductions with trees where the arc set A , as defined in the previous section, forms an arborescence. At the end of the section, we briefly investigate the consequences when this is not the case.

4.5.1 Objective Value

To investigate the problem more deeply, we start with reformulating the LP in the standard form. We use the following notation:

Definition 4.33. Let $\chi \in \{0, 1\}^{A \times V}$ be the matrix with

$$\chi_{av} = \begin{cases} 1 & \text{if } v \in T_a, \\ 0 & \text{otherwise,} \end{cases}$$

describing whether arc a points towards vertex v or not. We extend the notation of σ in terms of arcs with $\sigma = (\sigma_a)_{a \in A} \in \mathbb{R}_{\geq 0}^A$ where $\boldsymbol{\sigma}_a := \sigma(T_a)$. Let

$$\mathbf{M} := \left[\begin{array}{c|cc} & \chi_{A_{\text{in}},*} & \\ \hline \mathbb{1}_A & & \\ & \chi_{A_{\text{out}},*} & \\ \hline & -\chi_{A_{\text{in}},*} & \\ \mathbb{1}_A & & \\ & -\chi_{A_{\text{out}},*} & \end{array} \right], \quad \mathbf{b} := \left[\begin{array}{c} \sigma_{A_{\text{in}}} \\ \hline \sigma_{A_{\text{out}}} \\ \hline 2\vartheta_{\frac{1}{2}} \mathbb{1}_{A_{\text{in}}} - \sigma_{A_{\text{in}}} \\ \hline \sigma_{A_{\text{out}}} \end{array} \right],$$

with $M \in \{-1, 0, 1\}^{2|A| \times (1+|V|)}$, $b \in \mathbb{R}^{2|A|}$.

Using this definition, we can rewrite

$$\Theta = \{ \lambda \in \mathbb{R} \times \mathbb{R}^V : M\lambda \geq b, (0, \mathbb{1}_V^\top)\lambda = W \}$$

and the LP corresponding to the **STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES** can be equivalently formulated as

$$\begin{array}{l} \min (1, \mathbb{0}_V^\top)\lambda \\ \text{s.t. } M\lambda \geq b, \\ \quad (0, \mathbb{1}_V^\top)\lambda = W, \\ \quad \lambda \in \mathbb{R} \times \mathbb{R}^V. \end{array} \tag{4.7}$$

The rows of the matrix χ are the characteristic vectors of the vertex sets of the subtrees, whereas the columns are the characteristic vectors of the arcs on the paths from the root to each vertex. If

we sort the arcs in, e.g., breadth-first-search order, χ is a lower triangular square matrix with 1's on the main diagonal to which a zero column is attached corresponding to the unique root (as it is not part of any subtree). Obviously, the first half of the rows of M together with one row in the second half of the rows are linearly independent. Thus, we have $\text{rank}(M) = |V|$.

Now we can easily elaborate the dual problem:

Corollary 4.34. *The dual LP of (4.7) is*

$$\begin{array}{l} \max b^\top \alpha + W\beta \\ \text{s.t. } M^\top \alpha + (0, \mathbb{1}_V^\top)^\top \beta = (1, \mathbb{O}_V^\top)^\top, \\ \alpha \in \mathbb{R}_{\geq 0}^{2|A|}, \\ \beta \in \mathbb{R}. \end{array} \quad (4.8)$$

By just a few modifications, we can see that this dual problem has a very simple structure. We have

Lemma 4.35. *For A forming an arborescence and $\mathbf{c} := (\vartheta_{\frac{1}{2}} \mathbb{1}_{A_{\text{in}}}, \sigma_{A_{\text{out}}})$, an optimal solution λ^* to*

$$\begin{array}{l} \max c^\top \lambda \\ \text{s.t. } \mathbb{1}_A^\top \lambda = 1, \\ \lambda \in \mathbb{R}_{\geq 0}^A \end{array}$$

yields an optimal solution to (4.8) with $\alpha^ = \frac{1}{2}(\lambda^*, \lambda^*)^\top$ and $\beta^* = 0$.*

Proof. We show that (4.8) can be transformed into the above LP by removing redundant constraints and replacing the variables: With $\alpha = (\alpha^+, \alpha^-)^\top$ and $\alpha^+, \alpha^- \in \mathbb{R}_{\geq 0}^A$, we get for the matrix equation

$$\left[M^\top \left| \begin{array}{c} 0 \\ \mathbb{1}_V \end{array} \right. \right] \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \left[\begin{array}{c|c|c} \mathbb{1}_A^\top & \mathbb{1}_A^\top & 0 \\ \chi_A^\top & -\chi_A^\top & \mathbb{1}_V \end{array} \right] \begin{bmatrix} \alpha^+ \\ \alpha^- \\ \beta \end{bmatrix} = \begin{bmatrix} 1 \\ \mathbb{O}_V \end{bmatrix}.$$

This is equivalent to the constraints

$$\begin{aligned} \mathbb{1}_A^\top \alpha^+ + \mathbb{1}_A^\top \alpha^- &= 1, \\ \chi_v^\top \alpha^+ - \chi_v^\top \alpha^- + \beta &= 0 \quad \forall v \in V, \end{aligned}$$

where $\chi_v := \chi_{*,v}$ is the vector containing 1's for those arcs a which lie on the path from the root r to the vertex v . Due to $\chi_r = \mathbb{O}$, the second constraint for $v = r$ results in $\beta = 0$. Because of $\chi_v = \mathbf{e}_a$ for $a = (r, v)$, we have $\alpha_a^+ = \alpha_a^-$ for all $a \in A$ with $r \in a$. Furthermore, due to $\chi_w = \chi_v + \mathbf{e}_{vw}$ for all $(v, w) \in A$, we have

$$0 = \chi_w^\top \alpha^+ - \chi_w^\top \alpha^- = \chi_v^\top \alpha^+ - \chi_v^\top \alpha^- + \alpha_{vw}^+ - \alpha_{vw}^-$$

and therefore, inductively, $\alpha_a^+ = \alpha_a^-$ for all $a \in A$. Due to these relations, every feasible solution of (4.8) is of the form $(\alpha, \beta) = (\alpha^+, \alpha^+, 0)$ with $\alpha^+ \in \mathbb{R}_{\geq 0}^A$ holding

$$1 = \mathbb{1}_A^\top \alpha^+ + \mathbb{1}_A^\top \alpha^- = 2\mathbb{1}_A^\top \alpha^+.$$

By replacing $2\alpha^+ =: \lambda$, every feasible solution of (4.8) yields a feasible solution for the LP given in the lemma and vice versa. Furthermore, with

$$c := (\vartheta_{\frac{1}{2}} \mathbb{1}_{A_{\text{in}}}, \sigma_{A_{\text{out}}}),$$

we get for the objective value of a feasible solution

$$\begin{aligned} b^\top \alpha + W\beta &= \sum_{a \in A_{\text{in}}} \sigma_a \alpha_a^+ + \sum_{a \in A_{\text{out}}} \sigma_a \alpha_a^+ + \sum_{a \in A_{\text{in}}} (2\vartheta_{\frac{1}{2}} - \sigma_a) \alpha_a^- + \sum_{a \in A_{\text{out}}} \sigma_a \alpha_a^- \\ &= 2\vartheta_{\frac{1}{2}} \sum_{a \in A_{\text{in}}} \alpha_a^+ + 2 \sum_{a \in A_{\text{out}}} \sigma_a \alpha_a^+ \\ &= 2(\vartheta_{\frac{1}{2}} \mathbb{1}_{A_{\text{in}}}, \sigma_{A_{\text{out}}})^\top \alpha^+ \\ &= c^\top \lambda. \end{aligned}$$

Thus, the solutions have the same objective value and an optimal solution λ^* for the given LP provides an optimal solution for (4.8) with $\alpha^* = \frac{1}{2}(\lambda^*, \lambda^*)^\top$ and $\beta^* = 0$. \square

Due to the observed simple structure, we can now easily get the optimal value of the dual LP with

Lemma 4.36. *For A forming an arborescence, the optimal value of the LP of Lemma 4.35 is $\min\{\vartheta_0, \vartheta_{\frac{1}{2}}\}$.*

Proof. The LP is just a maximization over the unit $|A|$ -simplex and has the objective value $\max\{c_a : a \in A\}$. Due to $\vartheta_{\frac{1}{2}} > \sigma_a = \sigma(T_a)$ for all $a \in A_{\text{out}}$ by definition, we have two cases:

- a) $A_{\text{in}} = \emptyset$: Without inner arcs, we have $A = A_{\text{out}}$ and simply $c = \sigma_A$. Therefore, we have by Corollary 4.27

$$\max_{a \in A} c_a = \max_{a \in A} \sigma(T_a) = \vartheta_0 < \vartheta_{\frac{1}{2}}.$$

- b) $A_{\text{in}} \neq \emptyset$: For $a \in A_{\text{in}}$, we have $c_a = \vartheta_{\frac{1}{2}} \leq \sigma(T_a)$, hence $\max_{a \in A} c_a = \vartheta_{\frac{1}{2}}$ and

$$\vartheta_0 = \max_{a \in A} \sigma(T_a) \geq \vartheta_{\frac{1}{2}}. \quad \square$$

Returning to the original problem, we take advantage of the duality and have found the optimal value of our problem:

Theorem 4.37. *If G is a tree and A forms an arborescence, thus has a unique root, the optimal value of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM is $\vartheta_W = \min\{\vartheta_0, \vartheta_{\frac{1}{2}}\}$.*

Proof. From Lemma 4.36 with $a^* \in \arg \max\{c_a : a \in A\}$, we get the optimal primal value by the LP duality

$$\min\{\vartheta_0, \vartheta_{\frac{1}{2}}\} = c^\top e_{a^*} = b^\top \alpha^* + W\beta^* = (1, \mathbb{0}_V^\top) \lambda^* = \vartheta^*. \quad \square$$

Surprisingly, the objective value does only depend on the concrete tree structure in case ϑ_0 yields the optimal value. In turn, if it is equal to $\vartheta_{\frac{1}{2}}$, only the total σ -value and the total weight are decisive. As ϑ_0 is easy to evaluate, we can now also state:

Corollary 4.38. *If G is a tree and A forms an arborescence, we can find the optimal value of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|V|)$.*

Proof. Clear by the linear time calculation of ϑ_0 from Corollary 4.17 and the formula for $\vartheta_{\frac{1}{2}}$. \square

This result means that we can efficiently calculate the strength which is necessary to enforce the synchronization of the variables in the embedding. However, this is achieved under variable weights ω . Unfortunately, until now, we have not got any information on how to choose the ω 's to complete our embedding formulation. This is evaluated in the following section.

At this point, we like to remark the relation of our result with the one of V. Choi: Theorem 2 of [12] gives a bound for the coupling strength, transferred into our notation, of

$$\vartheta^* = \frac{|L| - 1}{|L|}(\sigma(V) - W),$$

where $L = \{v \in V : S(v) = \emptyset\}$ denotes the leaves of the tree. Apart from an isolated vertex, where no weight distribution is needed, all trees have $|L| \geq 2$. If the tree forms a path, in the quantum annealing context also called a *chain*, we have $|L| = 2$ and the given bound is equal to $\vartheta_{\frac{1}{2}}$. This means, in attendance of inner vertices, ϑ^* is indeed tight according to our problem formulation. In the remaining cases, it is easy to see that our optimization approach provides stronger bounds, improving the scaling factor and thus possibly also the success probability of the quantum annealer.

4.5.2 Straightforward Weights

Although we know there must exist a suitable weight for the previously found optimal value, we still need to find it. First of all, we can deduce for the set of feasible weights

Corollary 4.39. *With $\vartheta_W = \min\{\vartheta_0, \vartheta_{\frac{1}{2}}\}$, we have*

$$\Omega(\vartheta_W) = \left\{ \omega \in \mathbb{R}^V : \begin{aligned} \sigma(T_a) - \vartheta_W &\leq \omega(T_a) \leq \vartheta_W - \sigma(T_a) \quad \forall a \in A_{\text{out}}, \\ \omega(T_a) &= \sigma(T_a) - \vartheta_{\frac{1}{2}} \quad \forall a \in A_{\text{in}}, \\ \omega(V) &= W \end{aligned} \right\}.$$

Proof. From the proof of Lemma 4.36, we have $\vartheta_W = \vartheta_{\frac{1}{2}}$ for $A_{\text{in}} \neq \emptyset$. With

$$\omega^-(T_a, \vartheta_{\frac{1}{2}}) = \omega^+(T_a, \vartheta_{\frac{1}{2}})$$

for all $a \in A_{\text{in}}$, the inequalities in the formulation of Ω in Lemma 4.28 corresponding to A_{in} collapse to

$$\omega(T_a) = \sigma(T_a) - \vartheta_{\frac{1}{2}}.$$

For the case $A_{\text{in}} = \emptyset$ with $\vartheta_W = \vartheta_0$, there is nothing to show. \square

Remark: We have an analogous collapse of the inequalities for $a \in A_{\text{out}}$ in case of $A_{\text{in}} = \emptyset$, where the proof of Lemma 4.36 shows the optimal value of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES is ϑ_0 . With the problem formulation of Corollary 4.27, we can easily see that we require

$$\omega(T_{a^*}) = 0 \quad \forall a^* \in \arg \max_{a \in A} \sigma(T_a).$$

Note that this corollary is independent of A forming an arborescence or not. In case it does, we can analogously use the vertex notation with

$$\begin{aligned} \Omega(\vartheta_W) = \{ \omega \in \mathbb{R}^V : & \sigma(T_v) - \vartheta_W \leq \omega(T_v) \leq \vartheta_W - \sigma(T_v) \forall v \in V_{\text{out}}, \\ & \omega(T_v) = \sigma(T_v) - \vartheta_{\frac{1}{2}} \quad \forall v \in V_{\text{in}}, \\ & \omega(V) = W \}, \end{aligned}$$

since we have $V_{\text{in}} \neq \emptyset \Leftrightarrow A_{\text{in}} \neq \emptyset$.

By the above remark and the first remark following Definition 4.29, stating that we have $\omega^-(T_v, \vartheta) \leq 0 \leq \omega^+(T_v, \vartheta)$ for all $v \in V_{\text{out}}$ for $\vartheta \geq \vartheta_W$, it appears feasible to set the weight on the outer vertices to 0. By this the summed σ -values of the subtrees only become relevant on the first inner vertices. This could be understood as a ‘compression’ of the tree by shifting the σ ’s towards the root as illustrated in Figure 4.8. With the following theorem, we show that we can indeed find suitable weights for remaining vertices with this strategy.

Theorem 4.40. *If G is a tree and A forms an arborescence with root r , with $\vartheta_W = \min \{ \vartheta_0, \vartheta_{\frac{1}{2}} \}$ and*

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V_{\text{out}}, \\ \omega_v^* &= \sigma(T_v) - \vartheta_{\frac{1}{2}} - \sum_{s \in S_{\text{in}}(v)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) \quad \forall v \in V_{\text{in}}, \\ \omega_r^* &= W - \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}), \end{aligned}$$

we have $\omega^* \in \Omega(\vartheta_W)$ and $\omega^* \geq \mathbb{0}$.

Proof. With the above weight definitions, we have

$$\omega^*(T_v) = 0 \quad \forall v \in V_{\text{out}}.$$

Hence, due to the first remark following Definition 4.29, the inequalities concerning V_{out} of $\Omega(\vartheta_W)$, as given in Corollary 4.39, hold for the provided ω^* .

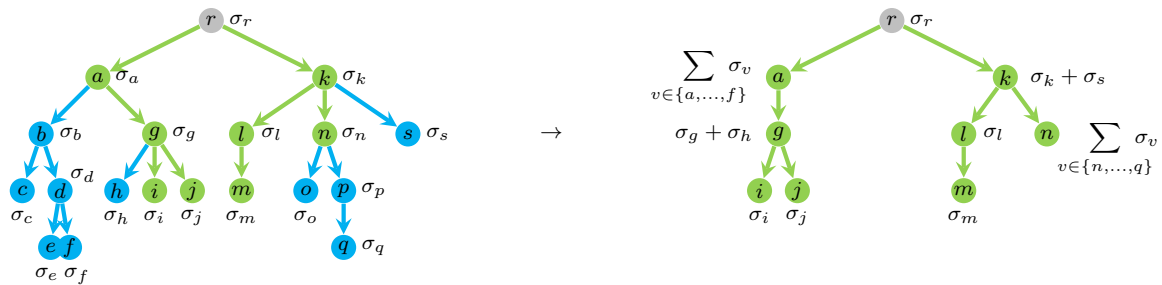


Figure 4.8: Compression of the tree to obtain only inner arcs and vertices

In case of $A_{\text{in}} = \emptyset$, we have $S_{\text{in}}(r) = \emptyset$ and it remains $\omega_r^* = W$ for the root r . Thus, we have $\omega^*(V) = W$. Since ω_r is unbounded, as it is not included in any inequality defining Θ in (4.6), this is a valid choice and we have $\omega^* \in \Omega(\vartheta_W)$. Furthermore, we also have $\omega^* = \mathbb{O} + W\mathbf{e}_r \geq \mathbb{O}$ with $W \geq 0$.

In case of $A_{\text{in}} \neq \emptyset$, with $\vartheta_W = \vartheta_{\frac{1}{2}}$ by the proof of Lemma 4.36, we get for the single vertex weights from the former formulations

$$\omega_v^* = \sigma(T_v) - \vartheta_{\frac{1}{2}}$$

for those $v \in V_{\text{in}}$ with $S_{\text{in}}(v) = \emptyset$, which are the leaves of the tree defined by the inner edges. Due to the additivity of ω , we also have

$$\omega^*(T_v) = \omega_v^* + \sum_{s \in S(v)} \omega^*(T_s) = \omega_v^* + \sum_{s \in S_{\text{in}}(v)} \omega^*(T_s) + \sum_{s \in S_{\text{out}}(v)} \omega^*(T_s) \quad \forall v \in V. \quad (\text{i})$$

With $\omega^*(T_v) = 0$ for all $v \in V_{\text{out}}$, this results in

$$\omega^*(T_v) = \omega_v^* = \sigma(T_v) - \vartheta_{\frac{1}{2}} \quad \forall v \in V_{\text{in}} \text{ with } S_{\text{in}}(v) = \emptyset.$$

By recursively resolving the subtree sums from the leaves upwards to the root of the arborescence by inserting the vertex weights given in the lemma in (i), we get

$$\begin{aligned} \omega^*(T_v) &= \sigma(T_v) - \vartheta_{\frac{1}{2}} - \sum_{s \in S_{\text{in}}(v)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) + \sum_{s \in S_{\text{in}}(v)} \omega^*(T_s) \\ &= \sigma(T_v) - \vartheta_{\frac{1}{2}} = \omega^-(T_v, \vartheta_{\frac{1}{2}}) = \omega^+(T_v, \vartheta_{\frac{1}{2}}) \end{aligned}$$

for all $v \in V_{\text{in}}$. Hence, the equalities of Corollary 4.39 concerning the inner vertices are respected, too.

Note that (i) also holds for the case $v = r$ with $T_r = V$. Thus, we further have

$$\begin{aligned} \omega^*(T_r) &= \omega^*(V) = \omega_r^* + \sum_{s \in S(r)} \omega^*(T_s) \\ &= W - \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) + \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) = W \end{aligned}$$

and ω^* as given in the lemma yields a feasible solution with $\omega^* \in \Omega(\vartheta_W)$.

It remains to show that $\omega^* \geq \mathbb{O}$ also holds for the vertices in $V_{\text{in}} \neq \emptyset$ and the root. For $v \in V_{\text{in}}$ with $S_{\text{in}}(v) = \emptyset$, we have $\omega_v^* = \sigma(T_v) - \vartheta_{\frac{1}{2}} \geq 0$ due to the definition of the inner vertices with $\sigma(T_v) \geq \vartheta_{\frac{1}{2}}$. With $\sigma \geq \mathbb{O}$ and $\vartheta_{\frac{1}{2}} > 0$, we have

$$\begin{aligned} \omega_v^* &= \sigma(T_v) - \vartheta_{\frac{1}{2}} - \sum_{s \in S_{\text{in}}(v)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) \\ &= \sigma_v + \sum_{s \in S_{\text{out}}(v)} \sigma(T_s) + \vartheta_{\frac{1}{2}} (|S_{\text{in}}(v)| - 1) \geq 0 \end{aligned}$$

for all $v \in V_{\text{in}}$ with $|S_{\text{in}}(v)| \geq 1$. By

$$\begin{aligned} \omega_r^* &= W - \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) \\ &= W - \sigma(V) + \sigma_r + \sum_{s \in S_{\text{out}}(r)} \sigma(T_s) + \vartheta_{\frac{1}{2}} |S_{\text{in}}(r)| \\ &= \sigma_r + \sum_{s \in S_{\text{out}}(r)} \sigma(T_s) + \vartheta_{\frac{1}{2}} (|S_{\text{in}}(r)| - 2), \end{aligned}$$

we also see that $\omega_r^* \geq 0$ for $|S_{\text{in}}(r)| \geq 2$. Since we need $|S_{\text{in}}(r)| \geq 1$ for $V_{\text{in}} \neq \emptyset$, the remaining case is $|S_{\text{in}}(r)| = 1$, for which let $S_{\text{in}}(r) = \{s\}$. Then with $\frac{1}{2}\sigma(V) - \sigma(T_s) \geq 0$, we have

$$\omega_r^* = W - \sigma(T_s) + \vartheta_{\frac{1}{2}} = \frac{1}{2}W - \sigma(T_s) + \frac{1}{2}\sigma(V) \geq 0$$

and thus $\omega^* \geq \mathbb{0}$. □

By this we have found an optimal solution for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES and, with these explicit formulas, it can also be calculated easily in the sense that

Corollary 4.41. *If G is a tree and A forms an arborescence, we can find an optimal solution for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|V|)$.*

Proof. By Corollary 4.38, we can find ϑ_W in linear time. Furthermore, the construction of the arborescence can also be done in linear time in the number of vertices by calculating the σ -sums of the subtrees starting on the leaves and stepping towards the root following a reversed depth first search. Analogously, we can calculate the corresponding weights from this. □

We have

$$\|\omega^*\|_\infty = \max_{v \in V} \omega_v^* = \max_{v \in V_{\text{in}}} \omega_v^*$$

for ω^* of Theorem 4.40. Thus, we can easily check whether we have already found an optimal solution for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES, too. Additionally, we see from this theorem that we can always find a non-negative weighting for a positive total weight. Thus, a restriction to $\omega \geq \mathbb{0}$ for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES is feasible.

4.5.3 Second-level Weight Optimization

In the previous section, we have constructed a weighting for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES straightforwardly using the optimal value ϑ_W . In this section, we evolve optimal weights for this problem in terms of their largest absolute value. Thus, we introduce another optimization level, where we only investigate the weights for the given optimal ϑ_W . By this we can already gain further knowledge about the full problem version: Only if we can find weights $\omega \in \Omega(\vartheta_W)$ which fulfil $\|\omega\|_\infty - \gamma \leq \vartheta_W$, the optimal value for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES is also ϑ_W .

Improved Weights

If we have not achieved $\vartheta_W + \gamma \geq \|\omega^*\|_\infty$ for ω^* of Theorem 4.40, the restriction to set the weights of the outer vertices to 0 might be too strict: There is some scope to increase the $\omega^*(T_v)$'s for $v \in V_{\text{out}}$ and shift some weight away from the inner vertices. This principle of opening the ‘compression’ to the next outer vertices is illustrated in Figure 4.9. We can indeed show that this allows an improvement of the maximal weight.

With the following lemma, we first construct the corresponding weights, where we highlight the differences to Theorem 4.40 in blue:

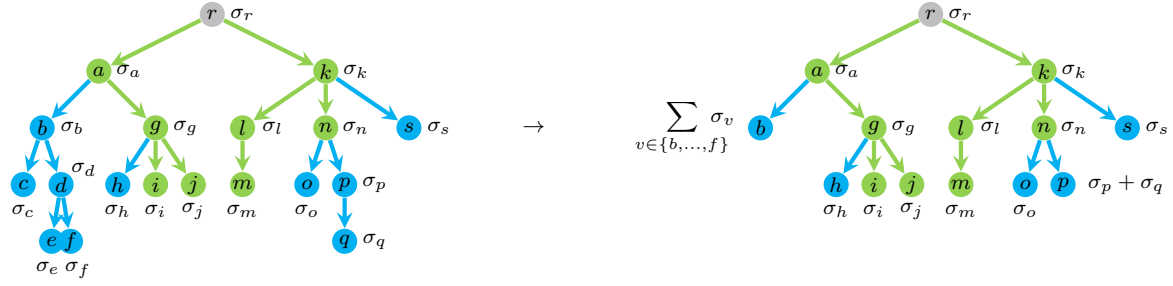


Figure 4.9: Compression of the tree keeping the next outer vertices

Lemma 4.42. *If G is a tree and A forms an arborescence with root r , for $\vartheta_W = \min \{\vartheta_0, \vartheta_{\frac{1}{2}}\}$ and $\omega^* \in \mathbb{R}^V$ with*

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}, \\ \omega_v^* &= \vartheta_W - \sigma(T_v) & \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\}, \\ \omega_v^* &= \sigma(T_v) - \vartheta_{\frac{1}{2}} - \sum_{s \in S_{\text{in}}(v)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) - \sum_{s \in S_{\text{out}}(v)} (\vartheta_{\frac{1}{2}} - \sigma(T_s)) & \forall v \in V_{\text{in}}, \\ \omega_r^* &= W - \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) - \sum_{s \in S_{\text{out}}(r)} (\vartheta_W - \sigma(T_s)), \end{aligned}$$

we have $\omega^* \in \Omega(\vartheta_W)$.

Proof. The validation of the vertex weights follows the one of Theorem 4.40: Due to

$$\omega^*(T_v) = 0 \quad \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}$$

and

$$\omega^*(T_v) = \omega_v^* = \vartheta_W - \sigma(T_v) \quad \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\},$$

the inequalities

$$\sigma(T_v) - \vartheta_W \leq \omega^*(T_v) \leq \vartheta_W - \sigma(T_v) \quad \forall v \in V_{\text{out}}$$

hold. Again by recursively exploiting the additivity of ω on the subtrees, but now also considering the contribution of the outer vertex weights, we get

$$\omega^*(T_v) = \sigma(T_v) - \vartheta_W = \omega^-(T_v, \vartheta_W) = \omega^+(T_v, \vartheta_W) \quad \forall v \in V_{\text{in}}.$$

If we have $V_{\text{in}} \neq \emptyset$, we know from the proof of Lemma 4.36 that $\vartheta_W = \vartheta_{\frac{1}{2}}$ and the equalities of Corollary 4.39 concerning the inner vertices thus also hold. Similarly, we can deduce $\omega^*(V) = W$ and therefore have $\omega^* \in \Omega(\vartheta_W)$. \square

Analogously to before, we can state about the computational time needed to get these weights:

Corollary 4.43. *The weights ω^* of Lemma 4.42 can be computed in $\mathcal{O}(|V|)$.*

Proof. The same argumentation as in the proof of Corollary 4.41 holds. \square

Furthermore, due to the shift of some weights to the next outer vertices by the subtraction of the highlighted terms, we have already achieved better weights in terms of the maximal weight value and can state

Lemma 4.44. For ω^* of Lemma 4.42 and $\tilde{\omega}^*$ of Theorem 4.40 with $\|\tilde{\omega}^*\|_\infty > \vartheta_W$, we have

$$\max_{v \in V} \omega_v^* \leq \max_{v \in V} \tilde{\omega}_v^*.$$

Proof. Due to $\tilde{\omega}^* \geq \mathbb{O}$, we know

$$\|\tilde{\omega}^*\|_\infty = \max_{v \in V} \tilde{\omega}_v^* > \vartheta_W.$$

By the first remark following Definition 4.29, we have $\vartheta_{\frac{1}{2}} - \sigma(T_v) \geq \vartheta_W - \sigma(T_v) \geq 0$ for all $v \in V_{\text{out}}$. Thus, on the one hand, we get $0 \leq \omega_v^* \leq \vartheta_W$ for all $v \in V_{\text{out}}$ and

$$\vartheta_W \geq \max_{v \in V_{\text{out}}} \omega_v^*.$$

On the other hand, we get $\omega_v^* \leq \tilde{\omega}_v^*$ for all $v \in V_{\text{in}} \cup \{r\}$ by the construction of the weights in Lemma 4.42, resulting in

$$\max_{v \in V} \tilde{\omega}_v^* \geq \max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* \geq \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v^*.$$

Combining all yields the above claim. \square

Unfortunately, in contrast to the weights given in Theorem 4.40, we do not necessarily have $\omega^* \geq \mathbb{O}$ for ω^* of Lemma 4.42. Due to the subtraction, there might now be inner vertices whose weights are negative, which is why we cannot state

$$\|\omega^*\|_\infty = \max_{v \in V} \omega_v^*$$

because there might be a vertex \tilde{v} with a large negative weight and

$$-\omega_{\tilde{v}} \geq \max_{v \in V} \omega_v^*.$$

However, with the following lemma, we see we can ‘repair’ a possibly appearing negative weight:

Theorem 4.45. If G is a tree and A forms an arborescence with root r , for $\omega^* \in \mathbb{R}^V$ of Lemma 4.42, there exists $\tilde{\omega}^* \in \Omega(\vartheta_W)$ with

- $\tilde{\omega}_v^* = 0$ for all $v \in V$ with $\omega_v^* < 0$ and
- $0 \leq \tilde{\omega}_v^* \leq \omega_v^*$ for all $v \in V$ with $\omega_v^* \geq 0$.

Thus, we have $\tilde{\omega}^* \geq \mathbb{O}$.

Proof. The difference of the ω^* ’s of Theorem 4.40 and Lemma 4.42 is a shift of some weight from inner vertices or the root to their outer successors. Therefore, only those vertices could get negative by shifting ‘too much’. Reducing the weight of the outer successors again properly results in $\tilde{\omega}^* \geq \mathbb{O}$. This is possible because we can freely choose the weight of such a successor s in $[\sigma(T_s) - \vartheta_W, \vartheta_W - \sigma(T_s)]$, being non-empty. By changing the weight of the inner predecessor v accordingly, the subtree weight remains constant with $\tilde{\omega}^*(T_v) = \omega^*(T_v)$ and the constraints stay fulfilled.

This means that $\tilde{\omega}^*$ can be constructed from ω^* of Lemma 4.42 with

$$\begin{aligned} \tilde{\omega}_v^* &= \omega_v^* && \forall v \in V_{\text{in}} \cup \{r\}, \omega_v^* \geq 0 \text{ and } \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}, \\ \tilde{\omega}_v^* &= 0 && \forall v \in V_{\text{in}} \cup \{r\}, \omega_v^* < 0, \\ \tilde{\omega}_s^* &= \omega_s^* - \varepsilon_s && \forall s \in S_{\text{out}}(v) \forall v \in V_{\text{in}} \cup \{r\}, \omega_v^* < 0 \end{aligned}$$

for some appropriate $\varepsilon \in \mathbb{R}_{\geq 0}^U$, $U := \{s \in S_{\text{out}}(v), v \in V_{\text{in}} \cup \{r\}, \omega_v^* < 0\}$, with $\varepsilon_s \leq \omega_s^*$ for all $s \in U$, and

$$\sum_{s \in S_{\text{out}}(v)} \varepsilon_s = |\omega_v^*| \quad \forall v \in V_{\text{in}} \cup \{r\}, \omega_v^* < 0.$$

Starting from the leaves, we get inductively for $v \in V_{\text{in}}$ with $\omega_v < 0$ that

$$\begin{aligned} \omega^*(T_v) &= \omega_v^* + \sum_{s \in S_{\text{out}}(v)} \omega_s^* + \sum_{s \in S_{\text{in}}(v)} \omega^*(T_s) \\ &= \omega_v^* + \sum_{s \in S_{\text{out}}(v)} (\tilde{\omega}_s^* + \varepsilon_s) + \sum_{s \in S_{\text{in}}(v)} \tilde{\omega}^*(T_s) \\ &= \omega_v^* + \underbrace{\sum_{s \in S_{\text{out}}(v)} \varepsilon_s}_{= 0 = \tilde{\omega}_v^*} + \sum_{s \in S(v)} \tilde{\omega}^*(T_s) \\ &= \tilde{\omega}^*(T_v) \end{aligned}$$

by assuming $\omega^*(T_s) = \tilde{\omega}^*(T_s)$ for all $s \in S_{\text{in}}(v)$. Thus, by this construction, we get $\tilde{\omega}^* \geq \mathbb{0}$ and $\tilde{\omega}_v^* \leq \omega_v^*$ for all $v \in V$ with $\omega_v^* \geq 0$ but still have $\tilde{\omega}^* \in \Omega(\vartheta_W)$. \square

Even with the required computational steps to repair the weights, we still have the linear time complexity:

Corollary 4.46. *The modified weights $\tilde{\omega}^*$ of Theorem 4.45 can be calculated in $\mathcal{O}(|V|)$.*

Proof. By iterating through the tree starting at the leaves analogously to the proof of Corollary 4.41, we can find the ε -values and get the repaired weights $\tilde{\omega}^*$ from ω^* , which can itself be found in linear time by Corollary 4.43. \square

Furthermore, with these corrected weights, we now have the desired equality

$$\|\tilde{\omega}^*\|_{\infty} = \max_{v \in V} \tilde{\omega}_v^*,$$

which means, if there are $v \in V$ with $\omega_v^* \leq 0$ for ω^* as constructed in Lemma 4.42, we can discard them in the further considerations of the maximum norm of the found ω^* . Additionally, we can see that we have achieved $\|\tilde{\omega}^*\|_{\infty} \leq \vartheta_W$, thus particularly $\|\tilde{\omega}^*\|_{\infty} \leq \vartheta_W + \gamma$, or only the inner vertices need to be checked whether their weight exceeds ϑ_W or not:

Corollary 4.47. *We have $\|\tilde{\omega}^*\|_{\infty} \leq \vartheta_W$ or*

$$\|\tilde{\omega}^*\|_{\infty} = \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v^* > \vartheta_W$$

for ω^ of Lemma 4.42 and the correspondingly modified $\tilde{\omega}^*$ of Theorem 4.45.*

Proof. Assume we have $\|\tilde{\omega}^*\|_{\infty} > \vartheta_W$. Then we get

$$\|\tilde{\omega}^*\|_{\infty} = \max_{v \in V} \tilde{\omega}_v^* = \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v^*$$

due to $\tilde{\omega}^* \geq \mathbb{0}$ and $\tilde{\omega}_v^* \leq \omega_v^* \leq \vartheta_W - \sigma(T_v) \leq \vartheta_W$ for all $v \in V_{\text{out}}$. Due to $\vartheta_W \geq 0$, the set $\{v \in V_{\text{in}} \cup \{r\} : \tilde{\omega}_v^* \geq 0\}$ is non-empty and we get

$$\max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* = \max_{\substack{v \in V_{\text{in}} \cup \{r\} \\ \tilde{\omega}_v^* \geq 0}} \tilde{\omega}_v^* = \max_{\substack{v \in V_{\text{in}} \cup \{r\} \\ \omega_v^* \geq 0}} \omega_v^* = \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v^*$$

because, by the construction in Theorem 4.45, we keep $\tilde{\omega}_v^* = \omega_v^*$ for all $v \in V_{\text{in}} \cup \{r\}$ which fulfil $\omega_v^* \geq 0$. \square

Using this relation, we can now establish a simple criterion which indicates whether we have found an optimal solution for the full problem:

Corollary 4.48. *If $\omega_v^* \leq \vartheta_W + \gamma$ holds for all $v \in V_{\text{in}} \cup \{r\}$ for ω^* of Lemma 4.42, we have $\vartheta_\diamond = \vartheta_W$ and $(\vartheta_W, \tilde{\omega}^*)$ is an optimal solution of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES with the modified $\tilde{\omega}^*$ of Theorem 4.45, hence*

$$(\vartheta_W, \tilde{\omega}^*) \in \arg \min \{(1, \mathbb{0})^\top \lambda : \lambda \in \Theta \cap \Phi^{-\gamma}\}.$$

Proof. From Corollary 4.47, we have $\|\tilde{\omega}^*\|_\infty \leq \vartheta_W + \gamma$ in this case and therefore $(\vartheta_W, \tilde{\omega}^*) \in \Phi^{-\gamma}$. Together with $(\vartheta_W, \tilde{\omega}^*) \in \arg \min \{(1, \mathbb{0})^\top \lambda : \lambda \in \Theta\}$, we have that $(\vartheta_W, \tilde{\omega}^*)$ is an optimal solution and the overall objective value of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES is ϑ_W . \square

Optimality

In the previous section, we have constructed improved weights for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES. Here, we show that there do not exist different weights that yield a smaller maximal value than those of Lemma 4.42. This result is achieved by several steps. First of all, we can show that for each inner vertex the weight is chosen minimally. For this we need the ‘not-yet-repaired’ weights of Lemma 4.42:

Lemma 4.49. *With ω^* of Lemma 4.42, we have*

$$\omega_v^* = \min \{\omega_v : \omega \in \Omega(\vartheta_W)\} \quad \forall v \in V_{\text{in}} \cup \{r\}.$$

Proof. From the proof of Lemma 4.42, we see that the ω -sums of the subtrees of outer successors get their maximal weight according to the upper bounds given by the inequalities describing $\Omega(\vartheta_W)$ in Corollary 4.39. This means

$$\omega^*(T_v) = \omega^+(T_v, \vartheta_W) = \max_{\omega \in \Omega(\vartheta_W)} \omega(T_v) \quad \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\}.$$

Furthermore, we have

$$\begin{aligned} \omega^*(T_v) &= \omega^-(T_v, \vartheta_W) = \min_{\omega \in \Omega(\vartheta_W)} \omega(T_v) \\ &= \omega^+(T_v, \vartheta_W) = \max_{\omega \in \Omega(\vartheta_W)} \omega(T_v) \quad \forall v \in V_{\text{in}}, \end{aligned}$$

due to the collapse of the inequalities in Corollary 4.39. With $T_r = V$, $\omega^*(V) = W$ and $\omega^-(T_r, \vartheta_W) = \omega^+(T_r, \vartheta_W) := W$, we can use the same notation also for the root r . Hence, for $v \in V_{\text{in}} \cup \{r\}$, we have

$$\begin{aligned} \omega_v^* &= \omega^*(T_v) - \sum_{s \in S(v)} \omega^*(T_s) \\ &= \min_{\omega \in \Omega(\vartheta_W)} \omega(T_v) - \sum_{s \in S(v)} \max_{\omega \in \Omega(\vartheta_W)} \omega(T_s) \\ &\leq \min_{\omega \in \Omega(\vartheta_W)} \left\{ \omega(T_v) - \sum_{s \in S(v)} \omega(T_s) \right\} \\ &= \min_{\omega \in \Omega(\vartheta_W)} \omega_v. \end{aligned}$$

As the reverse direction is trivial, the weights ω_v^* take on their minimal value for all $v \in V_{\text{in}} \cup \{r\}$. \square

Remember, although the weights from Lemma 4.42 can also be negative, we have adjusted this with Theorem 4.45. By this not all of the vertices might be chosen minimally anymore. In combination with the former lemma, we however see that we only have changed the maximal value of the weights of the inner vertices if all of them were negative before. If this is not the case, we have already found weights for the inner vertices whose maximal value cannot be chosen smaller:

Lemma 4.50. *For the modified $\tilde{\omega}^*$ of Theorem 4.45, we have*

$$\max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* = \max \left\{ 0, \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v \right\}.$$

Proof. With $\tilde{\omega}^* \in \Omega(\vartheta_W)$ and $\tilde{\omega}^* \geq \mathbb{0}$, we clearly have

$$\begin{aligned} \max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* &\geq \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v, \\ \max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* &\geq 0, \end{aligned}$$

which is combined

$$\max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* \geq \max \left\{ 0, \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v \right\}.$$

In the proof of Theorem 4.45 and by Lemma 4.49, we see that, with the corresponding ω^* of Lemma 4.42, we have

$$\tilde{\omega}_v^* = \max \{ 0, \omega_v^* \} = \max \left\{ 0, \min_{\omega \in \Omega(\vartheta_W)} \omega_v \right\}$$

for $v \in V_{\text{in}} \cup \{r\}$. Thus, we get the reverse inequality with

$$\begin{aligned} \max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* &= \max_{v \in V_{\text{in}} \cup \{r\}} \max \left\{ 0, \min_{\omega \in \Omega(\vartheta_W)} \omega_v \right\} \\ &= \max \left\{ 0, \max_{v \in V_{\text{in}} \cup \{r\}} \min_{\omega \in \Omega(\vartheta_W)} \omega_v \right\} \\ &\leq \max \left\{ 0, \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v \right\} \end{aligned}$$

since, for all $v \in V_{\text{in}} \cup \{r\}$ and for all $\omega \in \Omega(\vartheta_W)$, we have

$$\min_{\tilde{\omega} \in \Omega(\vartheta_W)} \tilde{\omega}_v \leq \omega_v \leq \max_{\tilde{v} \in V_{\text{in}} \cup \{r\}} \omega_{\tilde{v}}.$$

□

As we can exclude the case where the weights of Lemma 4.42 are negative for all inner vertices, we have $\|\omega^*\|_\infty \leq \vartheta_W$ in this case according to Corollary 4.47. We can thus deduce

Corollary 4.51. *If there exists at least one $v \in V_{\text{in}} \cup \{r\}$ with $\tilde{\omega}_v^* > 0$ for $\tilde{\omega}^*$ of Theorem 4.45, we have*

$$\max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* = \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v.$$

Proof. Clear from Lemma 4.50. □

With this result, only a single step is left to show the optimality of the constructed weights. For this we introduce a subset of the suitable weights $\Omega(\vartheta) = \{\omega : (\vartheta, \omega) \in \Theta\}$, where Θ , remember, denotes the set of feasible solutions of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES:

Definition 4.52. For $\vartheta \geq \vartheta_W$, let

$$\Omega^*(\vartheta) := \arg \min \{\|\omega\|_\infty : \omega \in \Omega(\vartheta)\}.$$

With this set of optimal weights for a given ϑ , we can now establish the desired optimality:

Lemma 4.53. *For $\|\tilde{\omega}^*\|_\infty > \vartheta_W$ with $\tilde{\omega}^*$ of Theorem 4.45, we have $\tilde{\omega}^* \in \Omega^*(\vartheta_W)$.*

Proof. With $\tilde{\omega}^* \in \Omega(\vartheta_W)$, we clearly have

$$\|\tilde{\omega}^*\|_\infty \geq \min \{\|\omega\|_\infty : \omega \in \Omega(\vartheta_W)\}.$$

From Corollaries 4.47 and 4.51, we have

$$\begin{aligned} \|\tilde{\omega}^*\|_\infty &= \max_{v \in V_{\text{in}} \cup \{r\}} \tilde{\omega}_v^* \\ &= \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V_{\text{in}} \cup \{r\}} \omega_v \\ &\leq \min_{\omega \in \Omega(\vartheta_W)} \max_{v \in V} \omega_v \\ &\leq \min_{\omega \in \Omega(\vartheta_W)} \|\omega\|_\infty, \end{aligned}$$

which results in the claimed equality. □

Note that we might have, but do not necessarily have, optimal weights in case of $\|\tilde{\omega}^*\|_\infty \leq \vartheta_W$. This is mainly caused by the outer vertices, whose weight is set to the upper bound on the whole subtree, which is already less than ϑ_W . Those weights could possibly be set to a lower value. This however is not relevant because we nevertheless have found an optimal solution for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES in this case.

All in all, we can now summarize the results with:

Theorem 4.54. *If G is a tree and A forms an arborescence with root r , with $\tilde{\omega}^*$ of Theorem 4.45, we have $\tilde{\omega}^* \in \Omega^*(\vartheta_W)$ or $\|\tilde{\omega}^*\|_\infty \leq \vartheta_W$.*

Proof. Clear from Lemma 4.53. □

While the latter means that $(\vartheta_W, \tilde{\omega}^*)$ is an optimal solution of the full problem straightforwardly, we cannot distribute the weights in a better way in the first case. Nevertheless, we might have $\|\tilde{\omega}^*\|_\infty \leq \vartheta_W + \gamma$ and have thus found an optimal solution of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. If this is not the case, we need to tackle the problem in a different way.

4.5.4 De-contraction

In this section, we briefly investigate the case where the arc set A does not form an arborescence in the tree G . We show that we can transfer the established results to this case by considering the contracted instance as introduced in Section 4.4.5. The weights we have found can be ‘de-contracted’ to obtain a weighting for the original instance. Note that $\tilde{\omega}^*$ in the following lemma does not necessarily need to be the one of Theorem 4.45.

Lemma 4.55. *Every optimal solution $(\tilde{\vartheta}^*, \tilde{\omega}^*)$ to the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES over the contracted instance yields an optimal solution $(\vartheta_{\frac{1}{2}}, \omega^*)$ to the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES over the uncontracted instance with*

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V \setminus (T_\ell \cup T_r), \\ \omega_v^* &= \tilde{\omega}_v^* & \forall v \in T_\ell \cup T_r \setminus \{\ell, r\}, \\ \omega_\ell^* &= \frac{1}{2}W - \tilde{\omega}^*(T_\ell \setminus \{\ell\}), \\ \omega_r^* &= \frac{1}{2}W - \tilde{\omega}^*(T_r \setminus \{r\}). \end{aligned}$$

Proof. With $\tilde{\omega}^*(T_{\tilde{a}}) = \omega^*(T_a)$ for all $\tilde{a} \in \tilde{A}$, the corresponding constraints are preserved. The arcs in $A_R \subseteq A_{\text{in}}$, corresponding to the edges on the path between ℓ and r , are not considered in the contracted instance. For such an $a \in A_R$, where we have $\sigma(T_a) = \frac{1}{2}\sigma(V)$, the inequality for all weights $\omega \in \Omega(\vartheta)$ for some ϑ is

$$\begin{aligned} \vartheta &\geq \vartheta_{\frac{1}{2}} + |\omega(T_a) + \vartheta_{\frac{1}{2}} - \sigma(T_a)| \\ &= \vartheta_{\frac{1}{2}} + |\omega(T_a) - \frac{1}{2}W|. \end{aligned}$$

Therefore, a trivial lower bound on the optimal value is $\vartheta_{\frac{1}{2}}$. With ω^* as given above, we have

$$\omega^*(T_\ell) = \omega^*(T_r) = \omega^*(T_a) = \frac{1}{2}W \quad \forall a \in A_R$$

and the absolute value on the right-hand side of the inequality vanishes. Thus, the inequality is fulfilled. Furthermore, we have

$$\omega^*(V) = \omega^*(T_\ell) + \omega^*(T_r) = W.$$

For the remaining arcs, $a = vw \in A \setminus (A_R \cup \tilde{A})$, we have $\sigma(T_a) = 0$ and therefore $a \in A_{\text{out}}$. With $\omega^*(T_a) = 0$ and $\vartheta_{\frac{1}{2}} > 0$, the corresponding constraints are fulfilled. Therefore, $(\vartheta_{\frac{1}{2}}, \omega^*)$ is a feasible solution with an objective value equal to the lower bound. □

This means, if A does not form an arborescence, the objective value of the STRENGTH-ONLY GAPPED WEIGHT DISTRIBUTION PROBLEM is always $\vartheta_{\frac{1}{2}}$.

Remembering the proof of Lemma 4.36, in case the contracted instance does not contain any inner arcs anymore, we get the optimal value $\tilde{\vartheta}_0$. Only in this case, the optimal value of the contracted instance might be less than the one of the uncontracted original instance, as we have $\tilde{\vartheta}_0 < \tilde{\vartheta}_{\frac{1}{2}} = \vartheta_{\frac{1}{2}}$. For the objective value ϑ_0 of the ZERO WEIGHT DISTRIBUTION PROBLEM over the uncontracted instance, we however have $\vartheta_0 = \frac{1}{2}\sigma(V) \geq \vartheta_{\frac{1}{2}}$. This means that Lemma 4.36 also holds if we do not have a unique root. Thus, we can now reformulate Corollary 4.41 without the restriction to an arborescence, which means we have solved the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES for all different cases:

Corollary 4.56. *If G is a tree, we can find an optimal solution for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|V|)$.*

Proof. Clear from Corollary 4.41 and the explicit formulas of Lemma 4.55. □

Proceeding analogously with the considerations for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES, do the preceding modifications preserve the optimality of the weights obtained from the contracted instance in the sense of Theorem 4.54? Or in other words, can we not achieve better weights than by ‘de-contracting’ $\tilde{\omega}^*$ of Theorem 4.45 according to Lemma 4.55? We see that in general the weights are already chosen optimally.

However, for the case $\tilde{A}_{\text{in}} = \emptyset$, we need to apply slight adjustments. As the contracted instance yields the optimal value $\tilde{\vartheta}_0 < \vartheta_{\frac{1}{2}}$ in this case, we have

$$\tilde{\omega}^*(T_v) = \omega^+(T_v, \tilde{\vartheta}_0) < \omega^+(T_v, \vartheta_{\frac{1}{2}})$$

for $v \in T_\ell \cup T_r = \tilde{V} \setminus \{\tilde{r}\}$ and the weights $\tilde{\omega}^* \in \mathbb{R}^{\tilde{V}}$ of Lemma 4.42. Thus, we loose the optimality argument when passing over to the uncontracted setting. Only by considering a feasible but non-optimal solution to the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES on the contracted instance, we can retain it. Therefore, we consider:

Lemma 4.57. *For $\tilde{A}_{\text{in}} = \emptyset$ and $\tilde{\omega}^* \in \mathbb{R}^{\tilde{V}}$ with*

$$\begin{aligned} \tilde{\omega}_v^* &= 0 & \forall v \in \tilde{V} \setminus (S(\tilde{r}) \cup \{\tilde{r}\}), \\ \tilde{\omega}_v^* &= \vartheta_{\frac{1}{2}} - \sigma(T_v) & \forall v \in S(\tilde{r}), \\ \tilde{\omega}_r^* &= W - \sum_{v \in S(\tilde{r})} (\vartheta_{\frac{1}{2}} - \sigma(T_v)), \end{aligned}$$

we have $\tilde{\omega}^ \in \Omega(\vartheta_{\frac{1}{2}})$.*

Proof. Due to $\tilde{A}_{\text{in}} = \emptyset$, we have $S(\tilde{r}) \subseteq \tilde{V}_{\text{out}}$ and

$$\omega^-(T_v, \vartheta_{\frac{1}{2}}) \leq \tilde{\omega}_v^* = \tilde{\omega}^*(T_v) \leq \omega^+(T_v, \vartheta_{\frac{1}{2}}) \quad \forall v \in \tilde{V} \setminus \{\tilde{r}\},$$

where we have equality in the last relation for $v \in S(\tilde{r})$. As the given weights also sum up to W , we have $\tilde{\omega}^* \in \Omega(\vartheta_{\frac{1}{2}})$. □

Remark: For these given weights, we do not necessarily have $\tilde{\omega}^* \geq \mathbb{O}$, but they can analogously be ‘repaired’ with Theorem 4.45. Furthermore, we observe that these weights also provide an optimal solution to the uncontracted instance with the same arguments as given in Lemma 4.55.

Now we can continue to consider the de-contracted instances, where we observe that the construction of Lemma 4.55 or Lemma 4.57 might introduce negative weights again. Therefore, we need to establish the non-negativity of the de-contracted weights analogously to Theorem 4.45, before we proceed with the optimality:

Theorem 4.58. *Let $\tilde{\omega}^*$ be a weighting for the contracted instance given by Theorem 4.45, respectively, Lemma 4.57 in case of $\tilde{A}_{in} = \emptyset$. If G is a tree and A does not form an arborescence, for ω^* obtained from $\tilde{\omega}^*$ by the construction of Lemma 4.55, there exists $\omega^+ \in \Omega(\vartheta_{\frac{1}{2}})$ with*

- $\omega_v^+ = 0$ for all $v \in V$ with $\omega_v^* < 0$ and
- $0 \leq \omega_v^+ \leq \omega_v^*$ for all $v \in V$ with $\omega_v^* \geq 0$.

Thus, we have $\omega^+ \geq \mathbb{O}$ and ω^+ can be found in $\mathcal{O}(|V|)$.

Proof. If we have $\omega^* \geq \mathbb{O}$, there is nothing to show. Since we have $\tilde{\omega}^* \geq \mathbb{O}$ and, by the construction of Lemma 4.55, we take over all weights apart from the one of the root of the contracted instance, we can only have $\omega_\ell^* < 0$ or $\omega_r^* < 0$ in case there exists a negative weight. As we further have $\omega_\ell + \omega_r = W \geq 0$, the negativity concerns only one root. Let this, w.l.o.g. due to the symmetry, be r .

We have

$$\begin{aligned} \omega_r^* &= \frac{1}{2}W - \sum_{s \in S(r) \cap T_r} \tilde{\omega}^*(T_s) \\ &= \frac{1}{2}W - \sum_{s \in S_{in}(r) \cap T_r} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) - \sum_{s \in S_{out}(r)} (\vartheta_{\frac{1}{2}} - \sigma(T_s)), \end{aligned}$$

which looks very similar to the root weight of Lemma 4.42. Analogously to the proof of Theorem 4.45, we can show that the negative weight is only caused by the contribution of the outer vertices on the right-hand side of the above equation. For this we first show

$$\frac{1}{2}W - \sum_{s \in S_{in}(r) \cap T_r} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) \geq 0, \quad (i)$$

where the left-hand side looks in turn similar to the weights of Theorem 4.40. Therefore, we can also adopt its proof: With $\sigma(T_r) = \frac{1}{2}\sigma(V)$ and $\sigma \geq \mathbb{O}$, we see by

$$\begin{aligned} \omega_r^* &= \frac{1}{2}W - \sum_{s \in S_{in}(r) \cap T_r} (\sigma(T_s) - \vartheta_{\frac{1}{2}}) \\ &= \frac{1}{2}W - \sigma(T_r) + \sigma_r + \sum_{s \in S_{out}(r)} \sigma(T_s) + \vartheta_{\frac{1}{2}} |S_{in}(r) \cap T_r| \\ &= \sigma_r + \sum_{s \in S_{out}(r)} \sigma(T_s) + \vartheta_{\frac{1}{2}} (|S_{in}(r)| - 1) \end{aligned}$$

that inequality (i) holds for $|S_{in}(r)| \geq 1$. Clearly, for $S_{in}(r) = \emptyset$, inequality (i) also holds with $W \geq 0$.

Now we can apply the same arguments as in the proof of Theorem 4.45 to repair the negative weight by reducing the weight on the outer neighbors and with this we can construct the desired non-negative weight $\omega^+ \in \Omega(\vartheta_{\frac{1}{2}})$. The runtime of Corollary 4.46 still applies. \square

Now we can deduce the analogous result for multiple roots:

Theorem 4.59. *If G is a tree and A does not form an arborescence, we have $\omega^+ \in \Omega^*(\vartheta_{\frac{1}{2}})$ or $\|\omega^+\|_{\infty} \leq \vartheta_W$ for ω^+ of Theorem 4.58.*

Proof. For the weights on the vertices in the subtrees T_r and T_ℓ , apart from ℓ and r themselves, we can follow the same proof steps as of Lemma 4.53 referring back to Lemma 4.49: The weights on the vertices cannot be reduced further because the subtree weights, subtracted in the inductive step, are chosen equal to their upper bound. Thus, they are also chosen optimally in the uncontracted instance.

By Corollary 4.39, we see the collapse of the inequalities for the inner arcs and we therefore need $\omega^+(T_a) = \frac{1}{2}W$ for all $a \in A_R$, resulting in $\omega^+(T_\ell) = \omega^+(T_r) = \frac{1}{2}W$. All remaining vertices in $R \setminus \{\ell, r\}$ as well as those attached to them can only be set to 0. By the additivity of ω^+ , the weights for the roots r and ℓ are directly given by the choice of the weights in T_ℓ and T_r and can thus also not be reduced further. Hence, all weights apart from those of the outer vertices are chosen optimally. Their weight is still lower or equal ϑ_W already and the above statement thus follows analogously to Theorem 4.54, respectively, Lemma 4.53. \square

This means we can state analogously to the section before: If the found weights ω^+ fulfil

$$\|\omega^+\|_{\infty} \leq \vartheta_W + \gamma,$$

we have also found an optimal solution for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES.

4.6 Full Problem on Trees

In this section, we deal with the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. As we have seen in the previous section, we can start by calculating the optimal value ϑ_W and corresponding second-level optimal weights ω^* of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES and check whether we have $\max_{v \in V} \omega_v^* \leq \vartheta_W + \gamma$ already. If this is not the case, we know we need to increase ϑ to enlarge the set of feasible weights $\Omega(\vartheta)$ and thus also possibly shift the set of optimal weights $\Omega^*(\vartheta)$.

In the following, we consider $\omega^*(\vartheta) \in \Omega^*(\vartheta)$ to be an arbitrary optimal weighting for a certain $\vartheta \geq \vartheta_W$. This means we have

$$\|\omega^*(\vartheta)\|_{\infty} = \min_{\omega \in \Omega(\vartheta)} \|\omega\|_{\infty}.$$

For instance, by Lemma 4.53, the weights $\omega^*(\vartheta_W)$ can be equal to $\tilde{\omega}^*$ of Theorem 4.45, but they do not need to be. Nevertheless, we have $\|\omega^*(\vartheta_W)\|_{\infty} = \|\tilde{\omega}^*\|_{\infty}$ for all choices of $\omega^*(\vartheta_W)$.

In the following, we only consider the case where $\|\omega^*(\vartheta_W)\|_{\infty}$ exceeds $\vartheta_W + \gamma = \min\{\vartheta_0, \vartheta_{\frac{1}{2}}\} + \gamma$. Remember, we further focus on the cases here where G is a tree and we have $0 < \vartheta_{\frac{1}{2}} < \frac{1}{2}\sigma(V)$ due to $\sigma(V) > W > 0$. The optimal value of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES is denoted by ϑ_{\diamond} .

4.6.1 Uniqueness

In this section, we derive some general results about the relation of ϑ and $\omega^*(\vartheta)$. These results are independent of whether the corresponding arborescence A has a unique root or not. First, we formally summarize the results of the previous section concerning their consequences for the full problem:

Corollary 4.60. *If we have $\|\omega^*(\vartheta_W)\|_\infty > \vartheta_W + \gamma$, the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES has an optimal value $\vartheta_\diamond > \vartheta_W$.*

This means we certainly know that the optimal value of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES ϑ_\diamond exceeds ϑ_W if the weights found in the previous section do not suffice. Furthermore, we can establish the following inverse proportionality relation:

Corollary 4.61. *For $\vartheta_W \leq \vartheta_1 \leq \vartheta_2$, we have $\|\omega^*(\vartheta_W)\|_\infty \geq \|\omega^*(\vartheta_1)\|_\infty \geq \|\omega^*(\vartheta_2)\|_\infty$.*

Proof. Clear from Corollary 4.30. □

Hence, by increasing ϑ , we can possibly decrease the maximal absolute value of the weights and thus close up both values. The search for the ‘sweet spot’ can be more formally expressed with:

Theorem 4.62. *The FULL WEIGHT DISTRIBUTION PROBLEM ON TREES is equivalent to the search for the (thus unique) ϑ with $\vartheta + \gamma = \|\omega^*(\vartheta)\|_\infty$.*

Proof. Clearly, for some $\tilde{\vartheta}$ with $\tilde{\vartheta} + \gamma = \|\omega^*(\tilde{\vartheta})\|_\infty$, we have

$$\begin{aligned} \tilde{\vartheta} &\geq \min \{ \vartheta \in \mathbb{R} : \vartheta + \gamma = \|\omega^*(\vartheta)\|_\infty \} \\ &\geq \min \{ \vartheta \in \mathbb{R} : \vartheta + \gamma \geq \|\omega^*(\vartheta)\|_\infty \} \\ &= \min \{ \vartheta : (\vartheta, \omega) \in \Theta \cap \Phi^{-\gamma} \} = \vartheta_\diamond. \end{aligned}$$

In turn, $\tilde{\vartheta} \geq \vartheta_\diamond$ via Corollary 4.61 also implies

$$\tilde{\vartheta} = \|\omega^*(\tilde{\vartheta})\|_\infty - \gamma \leq \|\omega^*(\vartheta_\diamond)\|_\infty - \gamma \leq \vartheta_\diamond,$$

resulting in the equality of $\tilde{\vartheta}$ and ϑ_\diamond . □

Using this equivalence, we can now reformulate

$$\begin{aligned} \vartheta_\diamond &= \min \{ \|\omega\|_\infty - \gamma : (\|\omega\|_\infty, \omega) \in \Theta \} \\ &= \min \{ \|\omega\|_\infty - \gamma : \omega \in \Omega(\|\omega\|_\infty) \}. \end{aligned}$$

In the following, we use this formulation to approach the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. Although Lemma 4.55 might not necessarily hold equivalently for this problem version, for the beginning, we also consider only unique roots.

Using the previous results, we can now already conclude the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES for a special case, namely when we have $A_{\text{in}} = \emptyset$. Remember, this means we always have a unique root and the optimal value for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES is ϑ_\diamond . This case is illustrated with the corresponding bounds on the subtree weights in Figure 4.10 on the next page.

By manipulating the formerly computed optimal weights for the strength-only problem, we can obtain the optimal value and the corresponding weights for the full one:

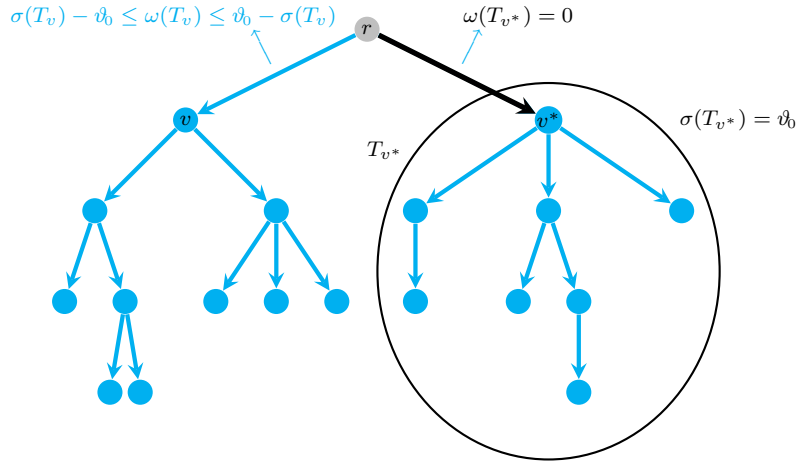


Figure 4.10: Arborescence with only outer vertices

Theorem 4.63. *If G is a tree and we have $A_{\text{in}} = \emptyset$ and $\|\omega^*\|_\infty > \vartheta_0 + \gamma$ with ω^* as constructed in Theorem 4.45, we have $\vartheta_\diamond = \vartheta_0 + \varepsilon^*$ with*

$$\varepsilon^* = \frac{\omega_r^* - \vartheta_0 - \gamma}{|S(r)| + 1}.$$

Proof. For $A_{\text{in}} = \emptyset$, we have $\vartheta_W = \vartheta_0$ according to the proof of Lemma 4.36 and, by the construction of Lemma 4.42, the weights ω^* are given with

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V \setminus (S(r) \cup \{r\}), \\ \omega_v^* &= \vartheta_0 - \sigma(T_v) & \forall v \in S(r), \\ \omega_r^* &= W - \sum_{v \in S(r)} \omega_v^*. \end{aligned}$$

By Corollary 4.47, we have $\|\omega^*\|_\infty = \omega_r^* > \vartheta_0 + \gamma$ and the application of Theorem 4.45 does not result in a different ω^* . By adding some $\varepsilon \geq 0$ to $\vartheta_0 + \gamma$, we can increase the upper bounds on the weights of the successors of the root. Let

$$\omega_v(\varepsilon) := \vartheta_0 + \varepsilon - \sigma(T_v) = \omega_v^* + \varepsilon \quad \forall v \in S(r)$$

be a function describing the weight in dependence of ε . This way, the weights still fulfil

$$\omega_v(\varepsilon) \leq \vartheta_0 + \varepsilon - \sigma(T_v) \leq \vartheta_0 + \varepsilon + \gamma.$$

Reducing the weight of the root accordingly by

$$\begin{aligned} \omega_r(\varepsilon) &= W - \sum_{v \in S(r)} \omega_v(\varepsilon) \\ &= W - \sum_{v \in S(r)} (\omega_v^* + \varepsilon) \\ &= W - \sum_{v \in S(r)} \omega_v^* - \varepsilon |S(r)| \\ &= \omega_r^* - \varepsilon |S(r)|, \end{aligned}$$

keeps

$$\sum_{v \in V} \omega_v(\varepsilon) = W$$

and we have $\omega(\varepsilon) \in \Omega(\vartheta_0 + \varepsilon)$. As $\omega_r(\varepsilon)$ is a decreasing linear function in ε , we still have

$$\|\omega(\varepsilon)\|_\infty = \omega_r(\varepsilon) \geq \vartheta_0 + \varepsilon + \gamma$$

for ε not being too large. Analogously to the proof of Lemma 4.49, the weights of the subtrees are set to their upper bound. Since $\omega_r(\varepsilon)$ does only depend on these weights, it cannot be reduced further for a fixed ε . Therefore, we also have $\omega(\varepsilon) \in \Omega^*(\vartheta_0 + \varepsilon)$.

Balancing the root weight with

$$\omega_r^* - \varepsilon^* |S(r)| = \vartheta_0 + \varepsilon^* + \gamma$$

results in the ε^* as stated in the lemma and

$$\|\omega^*(\vartheta_0 + \varepsilon^*)\|_\infty = \|\omega(\varepsilon^*)\|_\infty = \omega_r(\varepsilon^*) = \vartheta_0 + \varepsilon^* + \gamma.$$

Thus, according to Theorem 4.62, we have found the optimal value with $\vartheta_\diamond = \vartheta_0 + \varepsilon^*$. \square

Hence, the optimal solution of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES is given with $(\vartheta_0 + \varepsilon^*, \omega(\varepsilon^*))$ for $\omega(\varepsilon)$ as defined in the proof. Again, with these explicit formulas, we can easily see:

Corollary 4.64. *If G is a tree and we have $A_{\text{in}} = \emptyset$, we can find an optimal solution for the FULL WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|V|)$.*

Thus, the remaining case we need to consider is where we have $A_{\text{in}} \neq \emptyset$.

4.6.2 Non-negative Weights

In Section 4.5, we have seen that we can always find a suitable weighting for the optimal value ϑ_W of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES, where all weights are non-negative, which simplifies the consideration of the maximum norm. This holds even if those weights are required to be optimal, in the sense that their maximal absolute value is minimal. To approach the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES further, we start in this section with transferring this result to a more general case. By the following generalization of Theorem 4.45, we see that we can always restrict ourselves to non-negative weights:

Lemma 4.65. *If G is a tree and A forms an arborescence with root r , for fixed $\vartheta \geq \vartheta_W$ and $\omega \in \Omega(\vartheta)$, there exists $\tilde{\omega} \in \Omega(\vartheta)$ with*

- $\tilde{\omega}_v = 0$ for all $v \in V$ with $\omega_v < 0$ and
- $0 \leq \tilde{\omega}_v \leq \omega_v$ for all $v \in V$ with $\omega_v \geq 0$.

Thus, we have $\tilde{\omega} \geq \mathbb{0}$.

Proof. If we have $\omega \geq \mathbb{0}$ already, there is nothing to show. Given a vertex $v \in V$ with $\omega_v < 0$, we can assume that we have $\omega_w \geq 0$ for all $w \in T_v \setminus \{v\}$, as either the vertex is a leaf, which means there are no successors, or we have already handled all succeeding vertices when iterating through the tree starting at the leaves and moving upwards. We now need to differ between the following two cases for the vertex $v \in V$: We have either

A) $\omega(T_v) < 0$ or

B) $\omega(T_v) \geq 0$.

In the following, we show for both cases how to transform the weight of the subtree such that the resulting weighting, denoted by $\tilde{\omega}$, fulfils $\tilde{\omega}_w \geq 0$ for all $w \in T_v$, which means in particular $\tilde{\omega}_v \geq 0$, but we still have $\tilde{\omega} \in \Omega(\vartheta)$. Additionally, we construct $\tilde{\omega}$ such that we have in the end

$$\tilde{\omega}_v \leq \omega_v \quad \forall v \in V, \omega_v > 0.$$

The two cases are handled in the following way:

A) Let p be the predecessor of v . The case is illustrated in Figure 4.11(a). In the following, we show how to shift the whole negative weight of T_v up to ω_p without interfering with the inequalities defining $\Omega(\vartheta)$: As we have $\omega \in \Omega(\vartheta)$ with $\omega(T_v) < 0$, we know that $\omega^-(T_v, \vartheta) < 0$. Thus, with setting

$$\tilde{\omega}_w := 0 \quad \forall w \in T_v,$$

the constraints

$$\omega^-(T_w, \vartheta) < \tilde{\omega}(T_w) = 0 \leq \omega^+(T_w, \vartheta) \quad \forall w \in T_v$$

are still satisfied by the remark following Definition 4.29. Keeping the other vertices unchanged with

$$\tilde{\omega}_w := \omega_w \quad \forall w \in V \setminus (T_v \cup \{p\}),$$

we get

$$\tilde{\omega}(T_w) = \omega(T_w) \quad \forall w \in V, T_w \cap T_v = \emptyset.$$

With

$$\tilde{\omega}_p := \omega_p - |\omega(T_v)| = \omega_p + \omega(T_v)$$



Figure 4.11: Illustration of the two different cases in the iteration with vertices having a positive (black), a negative (red) or zero weight (white)

and the additivity of ω , we also get

$$\begin{aligned}\tilde{\omega}(T_p) &= \tilde{\omega}_p + \tilde{\omega}(T_v) + \sum_{s \in S(p) \setminus \{v\}} \tilde{\omega}(T_s) \\ &= \omega_p + \omega(T_v) + 0 + \sum_{s \in S(p) \setminus \{v\}} \omega(T_s) \\ &= \omega(T_p)\end{aligned}$$

and thus

$$\tilde{\omega}(T_w) = \omega(T_w) \quad \forall w \in V, T_p \subseteq T_w.$$

Hence, the shift of the weight does only influence the weighting of subtree T_v and not only the inequalities referring to all unchanged subtrees but also to all those subtrees defined by vertices in the arborescence above of p still hold, as well as $\tilde{\omega}(V) = W$. Therefore, we have $\tilde{\omega} \in \Omega(\vartheta)$. If we additionally have $\omega_p \geq |\omega(T_v)|$, we also get $0 \leq \tilde{\omega}_p \leq \omega_p$. Thus, we are done with the vertex v .

The procedure can be repeated for every possible sibling of v that also has a negative subtree weight. Hence, keeping p the same vertex as before, we can combine the cases with

$$\begin{aligned}\tilde{\omega}_w &:= 0 && \forall w \in T_v \quad \forall v \in S(p), \omega(T_v) < 0, \\ \tilde{\omega}_p &:= \omega_p + \sum_{\substack{v \in S(p) \\ \omega(T_v) < 0}} \omega(T_v)\end{aligned}$$

and $\tilde{\omega}_w := \omega_w$ for the remaining vertices w . Thus, analogously to the single case before, we still have $\tilde{\omega} \in \Omega(\vartheta)$. Additionally, we have achieved that

$$\tilde{\omega}_w = 0 \quad \forall w \in T_v \text{ with } \omega_w \leq 0,$$

while

$$0 \leq \tilde{\omega}_w \leq \omega_w \quad \forall w \in T_v \text{ with } \omega_w > 0$$

holds for all $v \in S(p)$ where $\omega(T_v) < 0$. By assuming the subtrees with $\omega(T_v) \geq 0$ for $v \in S(p)$ are handled already, we can extend the statement to all $v \in S(p)$.

In case of

$$\omega_p \geq - \sum_{\substack{v \in S(p) \\ \omega(T_v) < 0}} \omega(T_v),$$

we also achieved $0 \leq \tilde{\omega}_p \leq \omega_p$ and the whole subtree defined by p does not need to be considered further. If this is not the case, which means we do not have $\tilde{\omega}_p \geq 0$ in the end, we need to distinguish between two further cases:

- a) If we have $\tilde{\omega}(T_p) < 0$, we know that p cannot be equal to the root r . Otherwise we would have

$$W = \omega(V) = \tilde{\omega}(V) = \tilde{\omega}(T_p) < 0,$$

being a contradiction to the condition $W \geq 0$. Thus, we can continue to iterate upwards through the tree, where we now find case A) for p (the new vertex v) and its predecessor (the new p).

- b) If we have $\tilde{\omega}(T_p) \geq 0$, we can continue with case B) for p .

Both cases result in $\tilde{\omega}_p = 0$ and the iteration thus continues properly.

- B) The case is illustrated in Figure 4.11(b). Here, we show that the negative weight of v can be shifted downwards in the tree. Meanwhile, we can ignore all successors $s \in S(v)$ with $\omega(T_s) = 0$, which means we can assume $\omega(T_s) > 0$ for all $s \in S(v)$. There needs to be at least one such s due to $\omega_v < 0$ and

$$\omega(T_v) = \omega_v + \sum_{s \in S(v)} \omega(T_s) \geq 0.$$

In this construction, we show that we can decrease the weight of the subtrees T_s for $s \in S(v)$ such that we still have

$$\tilde{\omega}(T_s) \geq \max\{0, \omega^-(T_s, \vartheta)\}.$$

Let c_s denote the capacity of the subtree T_s with

$$\begin{aligned} c_s &= \omega(T_s) - \max\{0, \omega^-(T_s, \vartheta)\} \\ &= \omega(T_s) + \min\{0, \vartheta - \sigma(T_s)\}. \end{aligned}$$

Due to $\omega \in \Omega(\vartheta)$, we have $\omega(T_s) \geq \omega^-(T_s, \vartheta)$ and, since we know $\omega(T_s) \geq 0$, we also have $c_s \geq 0$ for all $s \in S(v)$. We show by contradiction that these capacities are sufficient to take all the negative weight of vertex v . Assume we have

$$\sum_{s \in S(v)} c_s < |\omega_v| = -\omega_v. \tag{i}$$

With

$$\begin{aligned} \sum_{s \in S(v)} c_s &= \sum_{s \in S(v)} \omega(T_s) + \sum_{s \in S(v)} \min\{0, \vartheta - \sigma(T_s)\} \\ &= \omega(T_v) - \omega_v + \sum_{\substack{s \in S(v) \\ \vartheta < \sigma(T_s)}} (\vartheta - \sigma(T_s)), \end{aligned}$$

inequality (i) is equivalent to

$$\omega(T_v) + \sum_{\substack{s \in S(v) \\ \vartheta < \sigma(T_s)}} (\vartheta - \sigma(T_s)) < 0.$$

Due to $\omega(T_v) \geq 0$, we have $\{s \in S(v) : \vartheta < \sigma(T_s)\} \neq \emptyset$ and thus

$$\begin{aligned} &\omega(T_v) + \sum_{\substack{s \in S(v) \\ \vartheta < \sigma(T_s)}} (\vartheta - \sigma(T_s)) \\ &= \omega(T_v) + \vartheta |\{s \in S(v) : \vartheta < \sigma(T_s)\}| - \sum_{\substack{s \in S(v) \\ \vartheta < \sigma(T_s)}} \sigma(T_s) \\ &\geq \omega(T_v) + \vartheta - \sigma(T_v) \\ &= \omega(T_v) - \omega^-(T_v, \vartheta), \end{aligned}$$

resulting in $\omega(T_v) < \omega^-(T_v, \vartheta)$, which is a contradiction to $\omega \in \Omega(\vartheta)$. Hence, we can shift the whole negative weight to successors according to the above capacities. If the weight of

the successor gets negative, that is, $\tilde{\omega}_s < 0$, we still have $\tilde{\omega}(T_s) \geq 0$ and we can thus apply case B) to s (the new v) now and iterate downwards the tree.

Once we have reached the leaves, we can apply the same procedure as before. As we know $\omega(T_s) = \omega_s \geq 0$, we see that the whole negative weight can be caught by the leaves' capacities. Thus, we are done with the whole subtree defined by v and can continue the iteration by checking the predecessor of v .

By this construction, we get the desired $\tilde{\omega} \in \Omega(\vartheta)$ with $\tilde{\omega} \geq \mathbb{0}$. □

With this result, we can now always use the equality $\|\omega\|_\infty = \max_{v \in V} \omega_v$. We can also easily see that the shift of the negative weights does not require expensive calculations:

Lemma 4.66. *The weights $\tilde{\omega}$ can be calculated in $\mathcal{O}(|V|)$.*

Proof. In the construction of Lemma 4.65, we iterate over all vertices starting at the leaves towards the root. Meanwhile, we can maintain all vertices with a positive weight in a depth-first-search manner until we meet the first one with a negative weight. In case the corresponding subtree fulfils case A), we set the weights of all vertices in the subtree to 0. For the following iteration steps, we can thus simply 'forget' all these vertices from the maintained ones.

In contrast, if we have case B), we need to enter the subtree again and iterate downwards to compensate the negative weight of the subtree root. This can be done by successively working through the maintained positive vertices until we have collected enough weight. As the weights of all these vertices, possibly apart from the last one, are also set to 0, they also do not need to be considered again, and we can remember the last vertex with capacity. If we need to enter the subtree again in a later iteration step fulfilling B), we can simply continue on the remembered vertex.

Thus, independently of how often we need to iterate downwards again, the vertices in the corresponding subtree are only considered once. This means, in total, we deal with every vertex only twice, which yields the above computational time. □

By the way this non-negative weight is constructed, the maximal absolute value is at least preserved. More formally, this means:

Corollary 4.67. *For $\vartheta \geq \vartheta_W$, $\omega \in \Omega(\vartheta)$ and the corresponding modified weights $\tilde{\omega}$ according to Lemma 4.65, we have*

$$\|\tilde{\omega}\|_\infty = \max_{v \in V} \tilde{\omega}_v \leq \max_{v \in V} \omega_v \leq \|\omega\|_\infty.$$

Proof. Since we have $W > 0$, there needs to exist at least one $v \in V$ with $\omega_v \geq 0$. By the relations of ω and $\tilde{\omega}$ given in Lemma 4.65, the claim follows. □

Since $\omega^*(\vartheta)$ was chosen to be optimal, this relation can only lead to:

Corollary 4.68. *For $\vartheta \geq \vartheta_W$, $\omega^*(\vartheta)$ and the corresponding modified $\tilde{\omega}^*$ according to Lemma 4.65, we have*

$$\|\tilde{\omega}^*\|_\infty = \|\omega^*(\vartheta)\|_\infty$$

and thus $\tilde{\omega}^* \in \Omega^*(\vartheta)$.

Proof. Clearly, if we had $\|\tilde{\omega}\|_\infty < \|\omega^*(\vartheta)\|_\infty$, the weights $\omega^*(\vartheta)$ would not be optimal. Thus, the statement directly follows from Corollary 4.67. \square

Finally summarizing the former results, we see that only the positive weights are decisive for the maximum norm:

Corollary 4.69. *For $\vartheta \geq \vartheta_W$, we have*

$$\|\omega^*(\vartheta)\|_\infty = \max_{v \in V} \omega_v^*(\vartheta).$$

Proof. If we had

$$\|\omega^*(\vartheta)\|_\infty = -\omega_{v^*}^* > \max_{v \in V} \omega_v^*$$

for some $v^* \in V$, we could ‘repair’ this negative weight by Lemma 4.65 and would thus have a lower maximum norm, which is a contradiction. \square

This on the one hand means that we can always restrict ourselves to non-negative weights, we hence have

Corollary 4.70. *For $\vartheta \geq \vartheta_W$, we have $\Omega^*(\vartheta) \cap \mathbb{R}_{\geq 0}^V \neq \emptyset$.*

On the other hand, if there appear negative weights when constructing an optimal solution, we can ‘ignore’ them for the moment and repair them later on. This is a decisive ingredient for the algorithm which we develop in the next section to deal with the remaining case where we have $A_{\text{in}} \neq \emptyset$.

4.6.3 Weight Calculation Algorithm

In the preceding part of this work, we have considered several special cases for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. In this section, we deal with the remaining case, where $\|\omega^*(\vartheta_W)\|_\infty - \gamma$ exceeds $\vartheta_W = \min\{\vartheta_0, \vartheta_{\frac{1}{2}}\}$, the optimal value of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES. Nevertheless, we still assume that G is a tree and the corresponding arc set A forms an arborescence and thus has a unique root r . In Theorem 4.63, we handled the case $A_{\text{in}} = \emptyset$. Hence, the remaining case is $A_{\text{in}} \neq \emptyset$, where we have $\vartheta_W = \vartheta_{\frac{1}{2}}$ according to the proof of Lemma 4.36 and thus $\|\omega^*(\vartheta_{\frac{1}{2}})\|_\infty - \gamma > \vartheta_{\frac{1}{2}}$. Further remember, due to $\sigma(V) > W > 0$, we have $0 < \vartheta_{\frac{1}{2}} < \frac{1}{2}\sigma(V)$.

General Principle

We construct the optimal value ϑ_\diamond for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES together with an optimal weighting $\omega_\diamond = \omega^*(\vartheta_\diamond)$ by defining a specific function ω depending on ϑ , which is denoted by $\omega(\vartheta) = (\omega_v(\vartheta))_{v \in V}$ in the following. We successively derive bounds on ϑ_\diamond by iterating through the vertices starting from the leaves towards the root. Let

$$\omega(\mathbf{T}_v, \vartheta) := \sum_{u \in T_v} \omega_u(\vartheta).$$

To have $\omega(\vartheta) \in \Omega(\vartheta)$, we fix

$$\begin{aligned}\omega_v(\vartheta) &:= 0 & \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}, \\ \omega_v(\vartheta) &:= \omega^+(T_v, \vartheta) = \vartheta - \sigma(T_v) & \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\},\end{aligned}$$

according to the bounds on the subtree weighting of the outer vertices like in the proof of Lemma 4.42. With this we have $\omega_v(\vartheta) = \omega(T_v, \vartheta)$ for all $v \in V_{\text{out}}$ with $p_v \in V_{\text{in}} \cup \{r\}$.

We derive an explicit formulation of $\omega_v(\vartheta)$ for the remaining inner vertices and the root by making use of the scope in the inequalities of $\Omega(\vartheta)$ in Lemma 4.28 for the inner vertices

$$\omega^-(T_v, \vartheta) = \sigma(T_v) - \vartheta \leq \omega(T_v, \vartheta) \leq \vartheta + \sigma(T_v) - 2\vartheta_{\frac{1}{2}} = \omega^+(T_v, \vartheta) \quad \forall v \in V_{\text{in}}.$$

With the additivity

$$\omega_v(\vartheta) = \omega(T_v, \vartheta) - \sum_{s \in S(v)} \omega(T_s, \vartheta),$$

we have

$$\omega_v^-(\vartheta) := \omega^-(T_v, \vartheta) - \sum_{s \in S(v)} \omega(T_s, \vartheta) \leq \omega_v(\vartheta) \leq \omega^+(T_v, \vartheta) - \sum_{s \in S(v)} \omega(T_s, \vartheta) =: \omega_v^+(\vartheta) \quad \forall v \in V_{\text{in}}.$$

This results in $\omega_v^-(\vartheta) = \omega_v^+(\vartheta) - 2\vartheta + 2\vartheta_{\frac{1}{2}}$ with $\omega_v^-(\vartheta_{\frac{1}{2}}) = \omega_v^+(\vartheta_{\frac{1}{2}})$.

Supposing we already know the ω -values for all the subtrees rooted at the successors of $v \in V_{\text{in}}$, we set

$$\omega_v(\vartheta) := \min \{ \vartheta + \gamma, \omega_v^+(\vartheta) \}$$

to always keep the weight for vertex v below $\vartheta + \gamma$ and its upper bound. With this single vertex weight, we can also describe the weighting for the whole subtree with

$$\begin{aligned}\omega(T_v, \vartheta) &= \omega_v(\vartheta) + \sum_{s \in S(v)} \omega(T_s, \vartheta) & (4.9) \\ &= \min \{ \vartheta + \gamma, \omega_v^+(\vartheta) \} + \sum_{s \in S(v)} \omega(T_s, \vartheta) \\ &= \min \left\{ \vartheta + \gamma + \sum_{s \in S(v)} \omega(T_s, \vartheta), \omega^+(T_v, \vartheta) \right\} \\ &= \vartheta + \min \left\{ \gamma + \sum_{s \in S(v)} \omega(T_s, \vartheta), \sigma(T_v) - 2\vartheta_{\frac{1}{2}} \right\}.\end{aligned}$$

In order to keep $\omega(\vartheta) \in \Omega(\vartheta)$, we further need to guarantee

$$\omega_v(\vartheta) \geq \vartheta + \gamma \stackrel{!}{\geq} \omega_v^-(\vartheta).$$

If we can achieve this for all $v \in V_{\text{in}}$, we have found the ϑ with $\vartheta + \gamma = \|\omega^*(\vartheta)\|_{\infty}$ according to Theorem 4.62. To evaluate the break points of the above conditions, we therefore need to solve

$$\begin{aligned}\omega_v^+(\vartheta) &= \vartheta + \gamma, \\ \omega_v^-(\vartheta) &= \vartheta + \gamma.\end{aligned}$$

Thus, the question is: How do the fixed points of $\omega_v^-(\vartheta) - \gamma$ and $\omega_v^+(\vartheta) - \gamma$ look like?

Leaves

We start with evaluating the fixed points at the leaves of the tree defined by the inner arcs. Due to the fixed weights on the attached subtrees, as those only contain outer vertices, we can derive explicit formulas: For some leaf ℓ of V_{in} , where $S_{\text{in}}(\ell) = \emptyset$, we have

$$\begin{aligned} \sum_{s \in S(\ell)} \omega(T_s, \vartheta) &= \sum_{s \in S_{\text{out}}(\ell)} (\vartheta - \sigma(T_s)) \\ &= |S_{\text{out}}(\ell)|\vartheta - \sum_{s \in S_{\text{out}}(\ell)} \sigma(T_s) \\ &= |S_{\text{out}}(\ell)|\vartheta - \sigma(T_\ell) + \sigma_\ell \end{aligned}$$

and therefore for the bounds

$$\begin{aligned} \omega_\ell^-(\vartheta) &= \sigma(T_\ell) - \vartheta - (|S_{\text{out}}(\ell)|\vartheta - \sigma(T_\ell) + \sigma_\ell) \\ &= -(|S_{\text{out}}(\ell)| + 1)\vartheta + 2\sigma(T_\ell) - \sigma_\ell \\ &=: \mathbf{s}_\ell^- \vartheta + \mathbf{c}_\ell^- \end{aligned}$$

and

$$\begin{aligned} \omega_\ell^+(\vartheta) &= \vartheta + \sigma(T_\ell) - 2\vartheta_{\frac{1}{2}} - (|S_{\text{out}}(\ell)|\vartheta - \sigma(T_\ell) + \sigma_\ell) \\ &= -(|S_{\text{out}}(\ell)| - 1)\vartheta + 2\sigma(T_\ell) - \sigma_\ell - 2\vartheta_{\frac{1}{2}} \\ &=: \mathbf{s}_\ell^+ \vartheta + \mathbf{c}_\ell^+, \end{aligned}$$

which are just linear functions in ϑ . By subtracting γ , we also only introduce a constant shift. The resulting functions have a single fixed point, thus unique solutions of $\omega_\ell^\mp(\vartheta) - \gamma = \vartheta$, if and only if the slope is not equal to 1. For a slope of 1, if the function is not the identity, there is no fixed point. Since we have $|S_{\text{out}}(\ell)| \geq 0$, the lower bound $\omega_\ell^-(\vartheta)$ always has a single fixed point, while, for the upper bound $\omega_\ell^+(\vartheta)$ to have a fixed point, we need $|S_{\text{out}}(\ell)| \neq 0$. Let ϑ_ℓ^- and ϑ_ℓ^+ be the corresponding fixed points. Then we have

$$\begin{aligned} \vartheta_\ell^- &= \frac{2\sigma(T_\ell) - \sigma_\ell - \gamma}{|S_{\text{out}}(\ell)| + 2} = \frac{c_\ell^- - \gamma}{1 - s_\ell^-} = \frac{c_\ell^+ + 2\vartheta_{\frac{1}{2}} - \gamma}{3 - s_\ell^+}, \\ \vartheta_\ell^+ &= \frac{2\sigma(T_\ell) - \sigma_\ell - 2\vartheta_{\frac{1}{2}} - \gamma}{|S_{\text{out}}(\ell)|} = \frac{c_\ell^+ - \gamma}{1 - s_\ell^+}, \end{aligned}$$

where the latter is omitted if we have $|S_{\text{out}}(\ell)| = 0$. This results in the constraint

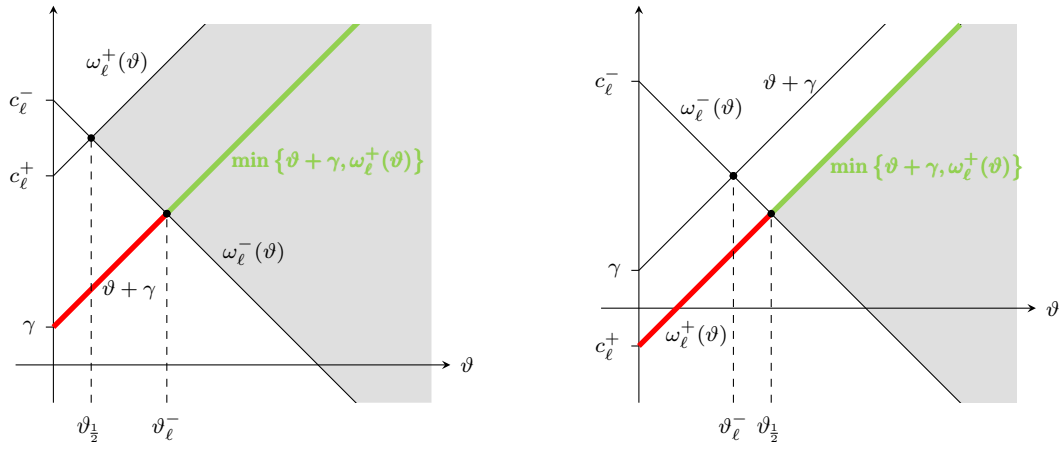
$$\vartheta \geq \max_{\substack{\ell \in V_{\text{in}} \\ S_{\text{in}}(\ell) = \emptyset}} \vartheta_\ell^-$$

and, in the following, we only need to consider functions of ϑ exceeding $\vartheta_{\frac{1}{2}}$ as well as a lower bound ϑ^- , which we initialize with the above maximum. In the following derivations, ϑ^- is maintained as the current best lower bound.

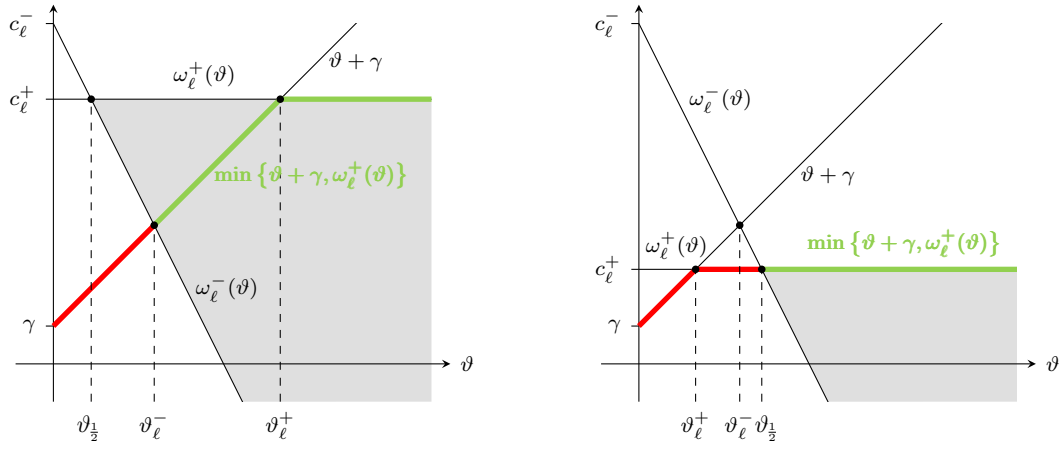
We set

$$\omega_\ell(\vartheta) := \min \{ \vartheta + \gamma, \omega_\ell^+(\vartheta) \} = \begin{cases} \vartheta + \gamma & \text{for } \vartheta < \vartheta_\ell^+, \\ \omega_\ell^+(\vartheta) & \text{otherwise,} \end{cases}$$

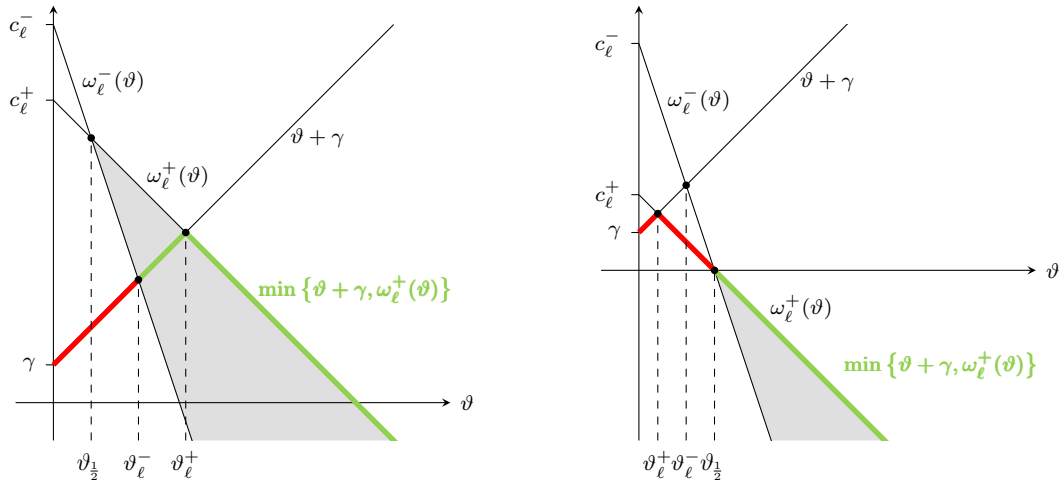
which is a piecewise linear, concave, continuous function. By setting ' $\vartheta_\ell^+ = \text{sign}(c_\ell^+ - \gamma) \cdot \infty$ ' in case of $|S_{\text{out}}(\ell)| = 0$, we can keep using the above definition because, in this case, we have just $\omega_\ell(\vartheta) = \vartheta + \gamma$ for $c_\ell^+ \geq \gamma$ or $\omega_\ell(\vartheta) = \vartheta + c_\ell^+$ otherwise. In Figure 4.12, several cases for the derived weight function for a leaf ℓ are illustrated.



(a) with $|S_{\text{out}}(\ell)| = 0$, thus $s_\ell^- = 1$ and $s_\ell^+ = -1$



(b) with $|S_{\text{out}}(\ell)| = 1$, thus $s_\ell^- = 2$ and $s_\ell^+ = 0$



(c) with $|S_{\text{out}}(\ell)| = 2$, thus $s_\ell^- = 3$ and $s_\ell^+ = 1$

Figure 4.12: Illustration of different cases of $\omega_\ell(\vartheta)$ for some leaf $\ell \in V_{\text{in}}$ with $S_{\text{in}}(\ell) = \emptyset$, and $\omega_\ell^\pm(\vartheta) = s_\ell^\pm \vartheta + c_\ell^\pm$ for $\vartheta_\ell^- > \vartheta_{\frac{1}{2}}$ (left) and $\vartheta_\ell^- < \vartheta_{\frac{1}{2}}$ (right)

We also obtain the weighting for the whole subtree with

$$\begin{aligned}\omega(T_\ell, \vartheta) &= \min \{ \vartheta + \gamma + |S_{\text{out}}(\ell)|\vartheta - \sigma(T_\ell) + \sigma_\ell, \omega^+(T_v, \vartheta) \} \\ &= \min \{ (|S_{\text{out}}(\ell)| + 1)\vartheta + \gamma - \sigma(T_\ell) + \sigma_\ell, \vartheta + \sigma(T_\ell) - 2\vartheta_{\frac{1}{2}} \},\end{aligned}$$

which is again a piecewise linear, concave, continuous function with the same break point ϑ_ℓ^+ as the function $\omega_\ell(\vartheta) - \gamma$ if it exists. The slopes and the additive constants can be calculated straightforwardly. It is easy to see that the slopes are larger than or equal to 1 for all linear pieces and, for a smaller ϑ , the slope is steeper. Therefore, $\omega(T_\ell, \vartheta)$ is also monotonically increasing.

Non-leaf Vertices

In the next step, we consider an arbitrary vertex $v \in V_{\text{in}}$ not being a leaf. How do $\omega_v^-(\vartheta)$ and $\omega_v^+(\vartheta)$ look like and what are the fixed points of $\omega_\ell^\pm(\vartheta) - \gamma$?

By iteration, we first show that $\omega(T_v, \vartheta)$ is a piecewise linear, concave, continuous and monotonically increasing function: We know that $\omega(T_s, \vartheta) = \vartheta - \sigma(T_s)$ for all $s \in S_{\text{out}}(v)$ are linear, concave, continuous and monotonically increasing functions and the iteration start is given by the former derivation for the leaves. Assuming $\omega(T_s, \vartheta)$ are piecewise linear, concave, continuous and monotonically increasing functions for all remaining successors $s \in S_{\text{in}}$, too, so is the sum

$$\sum_{s \in S(v)} \omega(T_s, \vartheta),$$

as all of these properties remain under summation. The number of break points amounts to at most the sum of the numbers of break points of the summands. Furthermore, also

$$\omega(T_v, \vartheta) = \vartheta + \min \left\{ \gamma + \sum_{s \in S(v)} \omega(T_s, \vartheta), \sigma(T_v) - 2\vartheta_{\frac{1}{2}} \right\}$$

is piecewise linear, concave, continuous and monotonically increasing but adds another break point. Additionally, the slopes of the linear parts are always larger than or equal to 1. With this

$$\begin{aligned}\omega_v^-(\vartheta) &= -\vartheta + \sigma(T_v) - \sum_{s \in S(v)} \omega(T_s, \vartheta), \\ \omega_v^+(\vartheta) &= \vartheta + \sigma(T_v) - 2\vartheta_{\frac{1}{2}} - \sum_{s \in S(v)} \omega(T_s, \vartheta)\end{aligned}$$

are piecewise linear, continuous, monotonically decreasing but convex functions for non-leaf vertices v and we always have only one fixed point for each. Let those be analogously defined to the former notations with ϑ_v^- and ϑ_v^+ . Then we can update the current best lower bound ϑ^- to $\max\{\vartheta^-, \vartheta_v^-\}$. If we have $\vartheta_v^+ > \vartheta^-$, we get an additional break point in

$$\begin{aligned}\omega_v(\vartheta) &:= \min \{ \vartheta + \gamma, \omega_v^+(\vartheta) \} \\ &= \begin{cases} \vartheta + \gamma & \text{for } \vartheta^- \leq \vartheta \leq \vartheta_v^+, \\ \omega_v^+(\vartheta) & \text{otherwise.} \end{cases}\end{aligned}$$

In some cases, we might also lose some of the break points, like illustrated in Figure 4.13.

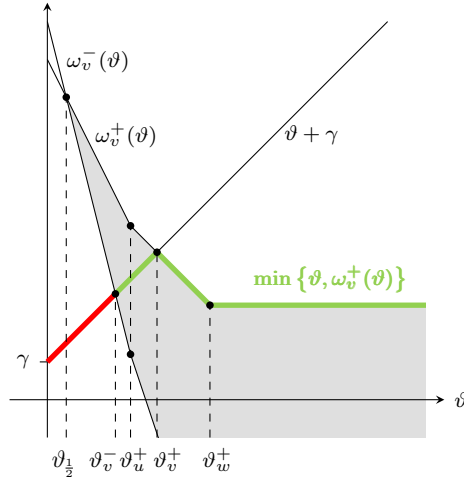


Figure 4.13: Illustration of $\omega_v(\vartheta)$ for $v \in V_{\text{in}}$ with $S_{\text{in}}(v) \neq \emptyset$ and $u, w \in T_v \cap V_{\text{in}}$

For a vertex $v \in V_{\text{in}}$, we then have

$$\begin{aligned}
 \sum_{s \in S(v)} \omega(T_s, \vartheta) &= \sum_{s \in S_{\text{out}}(v)} \omega_s(\vartheta) + \sum_{s \in S_{\text{in}}(v)} \omega(T_s, \vartheta) \\
 &= \sum_{s \in S_{\text{out}}(v)} (\vartheta - \sigma(T_s)) + \sum_{s \in S_{\text{in}}(v)} \omega(T_s, \vartheta) \\
 &= |S_{\text{out}}(v)|\vartheta - \sum_{s \in S_{\text{out}}(v)} \sigma(T_s) + \sum_{s \in S_{\text{in}}(v)} \omega(T_s, \vartheta) \\
 &= |S_{\text{out}}(v)|\vartheta - \sigma(T_v) + \sigma_v + \sum_{s \in S_{\text{in}}(v)} \sigma(T_s) + \sum_{s \in S_{\text{in}}(v)} \omega(T_s, \vartheta).
 \end{aligned}$$

By recursively computing the linear pieces for all subtrees, we finally reach the root r knowing the piecewise linear functions for all successors $s \in S(r)$. We have

$$\omega_r(\vartheta) = W - \sum_{S \in S(r)} \omega(T_s, \vartheta),$$

keeping $\omega(\vartheta) \in \Omega(\vartheta)$. As we also want to have $\omega_r(\vartheta) - \gamma \leq \vartheta$, we need to calculate the unique fixed point of this piecewise linear, continuous, monotonically decreasing, convex function, too. Let ϑ_r^- be the value for which we have $\omega_r(\vartheta_r^-) - \gamma = \vartheta_r^-$. We can finally update the lower bound to $\max\{\vartheta^-, \vartheta_r^-\}$. Let this value now be denoted by ϑ^- . As we have successively collected all conditions from every vertex, ϑ^- is the optimal value for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES in the end and we can get the desired weighting by inserting ϑ^- in the constructed ω -functions, that is, with $\omega(\vartheta^-)$.

After a short summary of the full algorithm in the next section, we provide a complete proof of the optimality in the subsequent section.

Summary

The algorithm presented in the previous sections starts with fixing the weights of the outer vertices to explicit linear functions in ϑ . The construction continues with the piecewise linear

weight functions in ϑ for the leaves of the tree formed by the inner arcs. Those functions have one break point each because they are formed by the maximum of ϑ and the upper bound on the vertex weight. In each step to the next level of the tree, we collect the pieces of the successors to combine them to the corresponding summed piecewise linear function and possibly attach another linear piece to continue with the same principle of keeping this sum below ϑ and its upper bound.

By iteratively collecting the pieces of linear functions for all inner vertices, we construct the weight functions for all vertices depending on a certain ϑ . By simultaneously updating the requirements on the lower bound, we ensure that we keep $\omega(\vartheta) \in \Omega(\vartheta)$. The final ‘sweet spot’ is then the best lower bound on ϑ which we can find. By inserting this value into the weight functions, we obtain the corresponding optimal weights.

To approach the weight functions computationally, we represent them in a specific format: As the considered piecewise linear, continuous functions are bounded to the left but unbounded to the right, we can identify such a function f with a family of ordered triples $((t_i, a_i, b_i))_{i=1, \dots, n}$ such that

$$f(\vartheta) = \begin{cases} a_1\vartheta + b_1 & \text{for } t_1 \leq \vartheta \leq t_2, \\ a_i\vartheta + b_i & \text{for } t_i \leq \vartheta \leq t_{i+1}, i = 2, \dots, n-1, \\ a_n\vartheta + b_n & \text{for } t_n \leq \vartheta. \end{cases}$$

Using this representation, we have

$$\begin{aligned} \omega(T_\ell, \vartheta) &\cong \begin{cases} ((\vartheta_\ell^-, 1, \gamma - \sigma(T_\ell) + \sigma_\ell)) & \text{if } s_\ell^- = 1 \wedge \vartheta_\ell^- > \vartheta_{\frac{1}{2}}, \\ ((\vartheta_{\frac{1}{2}}, 1, \sigma(T_\ell) - 2\vartheta_{\frac{1}{2}})) & \text{if } s_\ell^- > 1 \wedge \vartheta_\ell^- \leq \vartheta_{\frac{1}{2}}, \\ ((\vartheta_\ell^-, 2 - s_\ell, \gamma - \sigma(T_\ell) + \sigma_\ell), (\vartheta^+, 1, \sigma(T_\ell) - 2\vartheta_{\frac{1}{2}})) & \text{otherwise} \end{cases} \\ &=: \mathbf{P}_\ell \end{aligned} \tag{4.10}$$

for the leaves of the inner tree.

The sum of two piecewise linear functions, which is illustrated in Figure 4.14, can now be realized by considering the triples of the summands one after each other. We need to take over all break points but the first, as we update the current best lower bound, and sum up the entries for the slopes and the constant terms of the corresponding linear pieces separately to form the new pieces.

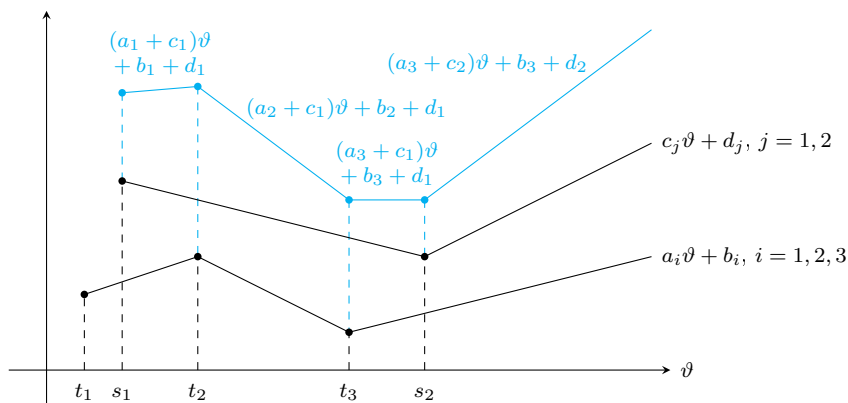


Figure 4.14: Illustration of the sum of two piecewise linear functions

In the pseudocode of Algorithm 4.1 on the next page, we provide a straightforward recursive implementation of the full described construction, with the corresponding helper functions such as `ADD` in Algorithm 4.2 on page 139. What we left out, for simplicity of the pseudocode, is that we also need to keep track of the U 's of line 34, respectively, line 30, for each vertex, as they provide the necessary pieces describing the weight function of the vertex.

Note that this code is not optimized concerning runtime but rather illustrates the overall principle. For instance, `GETFIXEDPOINT` could probably be implemented more efficiently with binary search or a specific data structure might improve `FILTERBYSTARTVALUE`. However, those functions are not critical in contrast to `ADD`. Instead of only pairwise addition, this method could be generalized to deal with the linear pieces of all successors at once. This is discussed in more detail in the following section.

Optimality

By the construction of the weights in the developed algorithm, we have iteratively considered all bounds on the subtree weights to finally obtain ϑ^- . We can indeed prove that this value provides the best weights in terms of the maximal value:

Lemma 4.71. *For G being a tree, A forming an arborescence and ϑ^- and the weight functions ω as constructed in Algorithm 4.1, we have*

$$\max_{v \in V} \omega_v(\vartheta^-) = \min_{\omega \in \Omega(\vartheta^-)} \max_{v \in V} \omega_v.$$

Proof. There are no further bounds given on the root r apart from the total weight sum. For simplicity of notation, let $\omega_r^-(\vartheta) := \omega_r(\vartheta)$. Then we can use the same notation for all $v \in V_{\text{in}} \cup \{r\}$. In total, we have

$$\vartheta^- = \max_{v \in V_{\text{in}} \cup \{r\}} \vartheta_v^-$$

with $\vartheta_v^- = \omega_v^-(\vartheta_v^-) - \gamma$ for all $v \in V_{\text{in}} \cup \{r\}$ and further $\vartheta^- \geq \vartheta_{\frac{1}{2}} > 0$.

We get the weight for each vertex $v \in V_{\text{in}}$ with

$$\omega_v(\vartheta^-) = \min \{ \vartheta^- + \gamma, \omega_v^+(\vartheta^-) \} \leq \vartheta^- + \gamma$$

and for each vertex $v \in V_{\text{out}}$ with

$$\begin{aligned} \omega_v(\vartheta^-) &= 0 < \vartheta^- + \gamma \quad \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}, \\ \omega_v(\vartheta^-) &= \vartheta^- - \sigma(T_v) < \vartheta^- + \gamma \quad \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\}. \end{aligned}$$

Furthermore, we have $\omega_v(\vartheta^-) \leq \vartheta^- - \gamma$ by construction. For $v^* \in \arg \max \{ \vartheta_v^- : v \in V_{\text{in}} \cup \{r\} \}$, we have

$$\vartheta^- = \vartheta_{v^*}^- = \omega_{v^*}^-(\vartheta^-) - \gamma.$$

Since we have $\omega_v^-(\vartheta) \leq \omega_v^+(\vartheta)$ for all $v \in V$ and $\vartheta \geq \vartheta_{\frac{1}{2}}$, we get

$$\omega_{v^*}(\vartheta^-) = \min \{ \vartheta^- + \gamma, \omega_{v^*}^+(\vartheta^-) \} = \vartheta^- + \gamma = \omega_{v^*}^-(\vartheta^-)$$

and thus

$$\omega_v(\vartheta^-) \leq \omega_{v^*}(\vartheta^-) \quad \forall v \in V.$$

Algorithm 4.1 Get Optimal Strength**Input:** $G = (V, E)$ tree, $\sigma \in \mathbb{R}_{\geq 0}^V$, $\vartheta_{\frac{1}{2}}$, $\gamma \in \mathbb{R}_{\geq 0}$, root r **Output:** $\vartheta_{\diamond} \in \mathbb{R}$

```

1:  $R \leftarrow \text{GETSUBTREEOMEGALINEARPIECES}(r)$  # get all linear pieces by recursion
2: return  $\text{GETFIXEDPOINT}(R)$  # return the current best lower bound

3: procedure  $\text{GETSUBTREEOMEGALINEARPIECES}(v)$ 
4:   if  $|S_{\text{in}}(v)| = \emptyset$  then # if the vertex is a leaf
5:     return  $P_v$  # we know the linear pieces by (4.10)
6:   for  $s \in S_{\text{in}}(v)$  do # get linear pieces of successors by recursion
7:      $P_s = ((t_i^s, a_i^s, b_i^s))_{i=1, \dots, k_s} \leftarrow \text{GETSUBTREEOMEGALINEARPIECES}(s)$  #  $P_s \cong \omega(T_s, \vartheta)$ 
8:    $\vartheta^- \leftarrow \max\{t_1^s : s \in S_{\text{in}}(v)\}$  # largest lower bound of all successors
9:    $\tilde{s} \leftarrow \arg \max\{t_1^s : s \in S_{\text{in}}(v)\}$  # corresponding successor
10:   $R \leftarrow P_{\tilde{s}}$  # start collecting the resulting pieces
11:  for  $s \in S_{\text{in}}(v) \setminus \{\tilde{s}\}$  do # sum up pieces of all successors
12:     $R \leftarrow \text{ADD}(R, P_s)$  # in the end have  $R \cong \sum_{s \in S_{\text{in}}(v)} \omega(T_s, \vartheta)$ 
13:   $O \leftarrow ((\vartheta^-, |S_{\text{out}}(v)|, -\sigma(T_v) + \sigma_v + \sum_{s \in S_{\text{in}}(v)} \sigma(T_s))$  # contribution of outer vertices
14:   $R \leftarrow \text{ADD}(P_s, O)$  # have  $R \cong \sum_{s \in S(v)} \omega(T_s, \vartheta)$ 
15:  if  $v = r$  then # if we consider the root we are done
16:    for  $(t_i, a_i, b_i) \in R$  do # correct pieces by total weight sum
17:       $(t_i, a_i, b_i) \leftarrow (t_i, -a_i, -b_i + W - \gamma)$  # in the end have  $R \cong \omega_r(\vartheta) - \gamma$ 
18:    return  $R$  # stop recursion and return root pieces
19:   $(L, U) \leftarrow (R, R)$  # set up lower/upper bound pieces of vertex
20:  for  $(t_i, a_i, b_i) \in L$  do # add difference to lower bound
21:     $(t_i, a_i, b_i) \leftarrow (t_i, -a_i - 1, -b_i + \sigma(T_v) - \gamma)$  # in the end have  $L \cong \omega_v^-(\vartheta) - \gamma$ 
22:  for  $(t_i, a_i, b_i) \in U$  do # add difference to upper bound
23:     $(t_i, a_i, b_i) \leftarrow (t_i, -a_i + 1, -b_i + \sigma(T_v) - 2\vartheta_{\frac{1}{2}} - \gamma)$  # in the end have  $U \cong \omega_v^+(\vartheta) - \gamma$ 
24:   $\vartheta_v^- = \text{GETFIXEDPOINT}(L)$  # fixed point of lower bound always exists
25:  if  $\vartheta_v^- > \vartheta^-$  then # if the new lower bound is better
26:     $\vartheta^- \leftarrow \vartheta_v^-$  # update current best lower bound
27:     $U \leftarrow \text{FILTERBYSTARTVALUE}(U, \vartheta^-)$  # update pieces of upper bound, too
28:   $\vartheta_v^+ = \text{GETFIXEDPOINT}(U)$  # fixed point or current best lower bound
29:   $B = ((\vartheta_v^+, 0, \gamma))$  #  $\gamma$  only needed for fixed point calculation
30:   $U \leftarrow \text{ADD}(U, B)$  # new start at  $\vartheta_v^+$ , remove  $\gamma$ ,  $U \cong \omega_v^+(\vartheta)$ 
31:  if  $\vartheta_v^+ > \vartheta^-$  then # if we have fixed point
32:     $((t_i, a_i, b_i))_{i=2, \dots, k+1} \leftarrow ((t_i, a_i, b_i))_{i=1, \dots, k}$  # shift indices
33:     $(t_1, a_1, b_1) \leftarrow (\vartheta^-, 1, \gamma)$  # introduce new piece with break point  $\vartheta_v^+$ 
34:     $U \leftarrow ((t_i, a_i, b_i))_{i=1, \dots, k+1}$  # combine pieces,  $U \cong \min\{\vartheta + \gamma, \omega_v^+(\vartheta)\}$ 
35:  return  $\text{ADD}(U, R)$  # return subtree upper bound,  $U \cong \omega(T_v, \vartheta)$ 

```

Algorithm 4.2 Helper Functions

```

1: procedure GETFIXEDPOINT( $((t_i, a_i, b_i))_{i=1, \dots, k}$ )
2:    $t_{k+1} \leftarrow \infty$  # define 'end point' of final interval
3:   for  $i = 1, \dots, k$  do # iterate over linear pieces
4:      $p_i \leftarrow \frac{b_i}{1-a_i}$  # calculate fixed point
5:     if  $t_i \leq p_i \leq t_{i+1}$  then # if fixed point is within the bounds
6:       return  $p_i$  # return the found fixed point
7:   return  $t_1$  # if no fixed point is found, use current best lower bound

8: procedure FILTERBYSTARTVALUE( $((t_i, a_i, b_i))_{i=1, \dots, k}, s$ )
9:    $\ell \leftarrow k$  # if all pieces have a start value  $\leq s$ , we use the last one
10:  for  $i = 1, \dots, k$  do # iterate over linear pieces
11:    if  $t_i > s$  then # if we find the first which has a larger start value
12:       $\ell \leftarrow \max\{0, i - 1\}$  # we start with this linear piece
13:    break # and can stop searching
14:   $t_\ell \leftarrow s$  # adopt the start value of the new first piece
15:  return  $((t_i, a_i, b_i))_{i=\ell, \dots, k}$  # return the filtered pieces

16: procedure ADD( $((t_i, a_i, b_i))_{i=1, \dots, k}, ((s_j, c_j, d_j))_{j=1, \dots, \ell}$ )
17:   $(i, j, m) \leftarrow (0, 0, 0)$  # initiate the counters
18:  while  $i \leq k \wedge j \leq \ell$  do # if we stay within the bounds of both counters
19:     $r_m \leftarrow (\max\{t_i, s_j\}, a_i + c_j, b_i + d_j)$  # collect the sum of the two single pieces
20:     $m \leftarrow m + 1$  # shift counter for resulting pieces
21:    if  $i + 1 > k$  then # if we have reached the end of the first family
22:       $j \leftarrow j + 1$  # shift the counter of the second one
23:    else if  $j + 1 > \ell$  then # if we have reached the end of the second family
24:       $i \leftarrow i + 1$  # shift the counter of the first one
25:    else # we have not reached the end
26:      if  $t_{i+1} \geq s_{j+1}$  then # if the next start value of the first exceeds the second
27:         $j \leftarrow j + 1$  # we handled the current linear piece of second family
28:      if  $t_{i+1} \leq s_{j+1}$  then # if the next start value of the second exceeds the first
29:         $i \leftarrow i + 1$  # we handled the current linear piece of first family
30:  return  $(r_n)_{n=1, \dots, m}$  # return the collected linear pieces

```

By the definition of ω^- , we can reformulate

$$\omega_{v^*}^-(\vartheta^-) = \min_{\omega \in \Omega(\vartheta^-)} \omega_{v^*}$$

and further deduce

$$\begin{aligned} \omega_{v^*}^-(\vartheta^-) &\leq \max_{v \in V} \min_{\omega \in \Omega(\vartheta^-)} \omega_v \\ &\leq \min_{\omega \in \Omega(\vartheta^-)} \max_{v \in V} \omega_v. \end{aligned}$$

This results in total in

$$\max_{v \in V} \omega_v(\vartheta^-) = \omega_{v^*}(\vartheta^-) = \vartheta^- + \gamma \leq \min_{\omega \in \Omega(\vartheta^-)} \max_{v \in V} \omega_v.$$

Because we have $\omega(\vartheta^-) \in \Omega(\vartheta^-)$ by the construction, we obtain the claimed equality. \square

In the construction, we have not considered the sign of the weights. Thus, there might be some vertices for which the found weight is negative. As we have seen before, those weights however are not decisive for the largest absolute value. Hence, we also derive the optimality of the found ϑ^- :

Theorem 4.72. *For G being a tree and A forming an arborescence, we have $\vartheta_\diamond = \vartheta^-$.*

Proof. According to Lemma 4.65, we can construct weights $\tilde{\omega} \in \Omega(\vartheta^-)$ with

$$\|\tilde{\omega}\|_\infty = \max_{v \in V} \omega_v(\vartheta^-) = \vartheta^- + \gamma.$$

By Lemma 4.71, we have

$$\|\tilde{\omega}\|_\infty = \min_{\omega \in \Omega(\vartheta^-)} \max_{v \in V} \omega_v \leq \min_{\omega \in \Omega(\vartheta^-)} \|\omega\|_\infty$$

and therefore $\tilde{\omega} \in \Omega^*(\vartheta^-)$. With Theorem 4.62, we have thus found the optimal value of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. \square

This means, with $(\vartheta^-, \tilde{\omega})$ for the weights $\tilde{\omega}$ of the proof, we have found an optimal solution of the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. Once we have constructed all piecewise linear functions yielding the weights of each vertex, we can simply insert a ϑ -value and obtain the suitable weights. Note that this can also be used to obtain even smaller weights by increasing ϑ even further. Thus, the user has the opportunity to adjust the ratio between the weights and the strength, which might be beneficial to improve the performance of the annealing machine.

However, to obtain these weight functions, we need to iteratively sum up piecewise linear functions while stepping through the tree. With the current implementation shown in the pseudocode, this results in the following runtime for our algorithm and thus for solving the full version of our problem:

Lemma 4.73. *If G is a tree and A forms an arborescence, we can find an optimal solution for the FULL WEIGHT DISTRIBUTION PROBLEM in $\mathcal{O}(|V|^3)$.*

Proof. To prove the stated runtime, we go through the procedure given in Algorithm 4.1: The filtering for the maximal start value only requires to check $|S(v)|$ values when considering vertex v . The for-loops working on R , U and L as well as the functions `GETFIXEDPOINT` and `FILTERBYSTARTVALUE` are only linear in the number of the combined linear pieces. Thus, the computation that is critical for the runtime is the `ADD` function, which we analyze in the following.

In the given algorithm, we have one break point in the linear pieces for the leaves of the tree formed by the inner arcs. In each step towards the root, we add the linear pieces, which in the worst case sums up the number of break points, too. Furthermore, we attach another linear piece for the corresponding vertex, which means we further increase the number of break points by 1. Thus, for a vertex v we have at maximum $|T_v|$ break points in the corresponding weight function.

The pairwise addition applied in the `ADD` function requires $\mathcal{O}(k+\ell)$ computational steps, where k and ℓ are the numbers of break points of the two summands. By adding another function with m pieces to the result, we have in total $\mathcal{O}((k+\ell) + (k+\ell+m))$ computational steps. Thus, by summing up the weight functions for all successors, which means $|S(v)|$ many, in each vertex v by the for-loop as shown, we handle the break points introduced by the first considered successor in all of the following additions again. The same holds for the subsequent functions and their break points until we reach the final one. As each of the successors might have a certain ratio of the $|T_v|$ vertices in its corresponding subtree, we have in total $\mathcal{O}(|S(v)||T_v|)$ computational steps.

This procedure is repeated for every vertex in V_{in} , which results in a runtime in $\mathcal{O}(f)$ with

$$f = \sum_{v \in V_{\text{in}}} |S(v)||T_v|,$$

where we clearly have $f \in \mathcal{O}(|V|^3)$. □

However, the `ADD` function as shown is implemented straightforwardly by simply adding the pieces of the next neighbor to the sum of the already handled neighbors. Using a better implementation might improve the runtime: The start times of all pieces of all neighbors form sorted lists, which need to be merged into one list. This can be done in $\mathcal{O}(\log(k)n)$, where k is the number of lists and n is the sum of the numbers of elements in all lists, because the lists could be merged pairwise and the same procedure could be repeated with the resulting lists, whose number has then reduced by half. This however only considers the start values but not the slopes and the constants of the linear pieces, which need to be summed up between the break points. We assume but do not check in detail that there is no further hidden complexity. Thus, the above f could possibly be changed to

$$f = \sum_{v \in V_{\text{in}}} \log(|S(v)||T_v|),$$

with which we could achieve a runtime of $\mathcal{O}(\log(|V|)|V|^2)$.

Note that, the way the weight functions are constructed, they respect all the inequalities defining $\Theta \cap \Phi^{-\gamma}$ when applying them to a $\vartheta \geq \vartheta^-$. This means they also provide feasible weights for the `STRENGTH-ONLY` and the `FULL WEIGHT DISTRIBUTION PROBLEM ON TREES` in this case.

Furthermore, although the algorithm is designed to deal with the remaining case as stated in the beginning of the section, we have not taken the condition $\|\omega^*(\vartheta_W)\|_\infty - \gamma \geq \vartheta_W = \min\{\vartheta_0, \vartheta_{\frac{1}{2}}\}$

into account. Thus, the algorithm is also applicable in the previously handled cases where we have $A_{\text{in}} \neq \emptyset$, however, has a worse runtime than the specific approaches.

The condition $A_{\text{in}} \neq \emptyset$ in turn is only responsible for starting the iteration with $\vartheta^- \geq \vartheta_{\frac{1}{2}}$. In case of $A_{\text{in}} = \emptyset$, we have a lower bound of only ϑ_0 with $\vartheta_0 < \vartheta_{\frac{1}{2}}$ according to the proof of Lemma 4.36. Furthermore, in this case, the algorithm would only provide a single non-trivial weight function, which is the one for the root r : As all neighbors of r are outer vertices, whose weights are set according to the upper bounds on the subtree, we directly apply the final step of the algorithm and get

$$\omega_r(\vartheta) = W - \sum_{s \in \mathcal{S}(r)} (\vartheta - \sigma(T_s)) \stackrel{!}{\leq} \vartheta + \gamma.$$

Rearranging the inequality for ϑ results in the bound

$$\vartheta \geq \frac{W + \sigma(V) - \sigma_r - \gamma}{|\mathcal{S}(r)| + 1}.$$

The right-hand side is indeed equal to $\vartheta_0 + \varepsilon^*$, the bound as established in Theorem 4.63, which can easily be shown by resolving ε^* . This means we can also ‘apply the algorithm’ in this case.

4.6.4 Adjusted Algorithm Including De-contraction

In this section, we investigate the case where the arc set A does not form an arborescence, which means we do not have a unique root but rather a set of ‘roots’ R . The edges connecting these roots form a path and the corresponding arc set A_R contains two arcs for each edge, pointing in the opposite directions. Let ℓ and r denote the endpoints of the path as introduced in Section 4.4.5.

By contracting the instance as described in Definition 4.32, we can get a first insight in the overall structure of the tree and, with Lemma 4.55 of Section 4.5.4, we can construct a first possible weighting. However, there is no guarantee for the optimality of the thus obtained values for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES. Therefore, we briefly describe a possible algorithm here showing how an optimal solution might be obtained.

General Principle

In the case of non-unique roots, where $A_R \neq \emptyset$, the additional arcs introduce further constraints with

$$\vartheta \geq \vartheta_{\frac{1}{2}} + \left| \vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a, \vartheta) \right|$$

for $a \in A_R$. Remember, those arcs are always inner arcs. With $\sigma(T_a) = \frac{1}{2}\sigma(V)$ and the additivity of ω , the above inequality is equivalent to

$$\frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) \leq \omega(T_a, \vartheta) \leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}),$$

respectively, for $a = vw$ to

$$\frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) - \sum_{\substack{s \in \mathcal{S}(v) \\ s \neq w}} \omega(T_s, \vartheta) \leq \omega_v(\vartheta) \leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) - \sum_{\substack{s \in \mathcal{S}(v) \\ s \neq w}} \omega(T_s, \vartheta).$$

This means, in contrast to the situation for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES, where the inequalities collapse to $\omega(T_a) = \frac{1}{2}W$ as shown in the proof of Lemma 4.55, we gain a scope of $\pm(\vartheta - \vartheta_{\frac{1}{2}})$ here, which can somehow be shifted between the roots. For the weights obtained by Lemma 4.55, there might be a large imbalance between the weights of the two roots ℓ and r , the endpoints of the path formed by the arcs in A_R . We are now possibly able to balance these weights to a certain degree and with this reduce the optimal value.

We can adjust the former algorithm as follows: We start with the iteration at the leaves of the tree formed by the inner arcs and step upwards until we have constructed the weight functions for all vertices in $V \setminus R$. As the arborescence is well defined for those vertices, this can be done in exactly the same way as explained before for the case of unique roots. Equivalently, we maintain the corresponding current best lower bound ϑ^- until this point.

Now we need to extend the algorithm to be able to handle the multiple roots: First of all, we always have arcs pointing towards the two roots ℓ and r and the given bounds therefore include

$$\begin{aligned}\omega_{\ell/r}^-(\vartheta) &= \frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) - \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta), \\ \omega_{\ell/r}^+(\vartheta) &= \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) - \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta),\end{aligned}$$

where the weight functions $\omega(T_s, \vartheta)$ for $s \in S(\ell/r) \cap T_{\ell/r}$ are already known. Again, we also restrict the weights $\omega_{\ell/r}(\vartheta)$ by $\vartheta + \gamma$ from above, which means in total

$$\omega_{\ell/r}^-(\vartheta) \leq \omega_{\ell/r}(\vartheta) \leq \min \{ \vartheta + \gamma, \omega_{\ell/r}^+(\vartheta) \}.$$

Hence, we need to guarantee analogously to before that

$$\vartheta \stackrel{!}{\geq} \omega_{\ell/r}^-(\vartheta) - \gamma.$$

The resulting fixed points ϑ_{ℓ}^- and ϑ_r^- of $\omega_{\ell}^-(\vartheta) - \gamma$, respectively, $\omega_r^-(\vartheta) - \gamma$ introduce additional lower bounds that need to be maintained with ϑ^- . Those can easily be calculated similarly to the single root bound.

Furthermore, we get analogously to (4.9)

$$\begin{aligned}\omega(T_{\ell/r}, \vartheta) &= \omega_{\ell/r}(\vartheta) + \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta) \\ &\leq \min \{ \vartheta + \gamma, \omega_{\ell/r}^+(\vartheta) \} + \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta) \\ &= \min \left\{ \vartheta + \gamma + \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta), \omega^+(T_{\ell/r}, \vartheta) \right\} \\ &= \vartheta + \min \left\{ \gamma + \sum_{s \in S(\ell/r) \cap T_{\ell/r}} \omega(T_s, \vartheta), \frac{1}{2}W - \vartheta_{\frac{1}{2}} \right\} \\ &=: \omega^{\leq}(T_{\ell/r}, \vartheta),\end{aligned}$$

being piecewise linear, concave, continuous and monotonic, just like the weight functions $\omega_{\ell/r}(\vartheta)$ themselves. While the latter are monotonically decreasing, the former are again increasing.

Note that we do not have the equality in the preceding relation like in (4.9). Assigning the largest possible value to the root weights according to their upper bounds might lead to exceeding the total weight. We need to balance the remaining weight within the new bounds optimally. How to do this depends on the number of roots $|R|$ or whether there are further vertices attached to the root nodes in the inner of the path formed by A_R .

Two Roots

In the case where we only have $R = \{\ell, r\}$, no further constraints on the subtrees apart from the already mentioned ones exist. However, we still need to ensure that the collected weights sum up to the total weight W , like in the final construction step for the unique root. If we only have the two roots ℓ and r , we therefore need

$$\omega(T_\ell, \vartheta) + \omega(T_r, \vartheta) = \omega(V, \vartheta) = W.$$

Hence, for the bounds defined in the previous paragraph, we require

$$\omega^{\leq}(T_\ell, \vartheta) + \omega^{\leq}(T_r, \vartheta) \geq W$$

to enforce that the upper bounds on the weighting actually suffice to sum up to the total weight. Since all parts of the left-hand side of the last inequality are known and their sum is again a piecewise linear, concave, continuous and monotonically increasing function, the corresponding equation has a unique solution. Let ϑ_W^- be the corresponding value. Thus, with $\vartheta \geq \vartheta_W^-$, we can find $\omega_\ell(\vartheta)$ and $\omega_r(\vartheta)$ such that $\omega(V, \vartheta) = W$ and we have $\omega(\vartheta) \in \Omega(\vartheta)$.

As ϑ_W^- introduces a further lower bound on the optimal value, we can finally update the lower bound to $\max\{\vartheta^-, \vartheta_\ell^-, \vartheta_r^-, \vartheta_W^-\}$, where ϑ^- was the maintained former best lower bound and is set to the new maximum. Analogously to before, we found ϑ^- as the optimal value for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES and we can get the desired weighting by inserting ϑ^- in the constructed ω -functions. However, only if ϑ_W^- dominates all other lower bounds, which means we have $\vartheta^- = \vartheta_W^-$, we can apply this also for the roots: In this case, we need

$$\omega(T_{\ell/r}, \vartheta_W^-) = \omega^{\leq}(T_{\ell/r}, \vartheta_W^-)$$

to achieve the total weight and therefore can directly obtain the root weights with

$$\omega_{\ell/r}^* = \omega_{\ell/r}(\vartheta_W^-) = \min\{\vartheta_W^- + \gamma, \omega_{\ell/r}^+(\vartheta_W^-)\}.$$

For $\vartheta^- > \vartheta_W^-$, we can in general not give explicit formulas for $\omega_\ell(\vartheta)$ and $\omega_r(\vartheta)$ and therefore not obtain $\omega_\ell(\vartheta^-)$ and $\omega_r(\vartheta^-)$ directly. We rather have to evaluate the found boundary functions: Applied to ϑ^- , they provide explicit bounds the weights need to comply with. For ω_ℓ^* and ω_r^* , denoting a possible choice for the weights of ℓ and r , we require

$$\omega_{\ell/r}^-(\vartheta^-) \leq \omega_{\ell/r}^* \leq \min\{\vartheta^- + \gamma, \omega_{\ell/r}^+(\vartheta^-)\}.$$

As the choice of the weight of one root influences the other due to the total weight sum

$$\omega_\ell^* + \omega_r^* = W - \omega(V \setminus \{\ell, r\}, \vartheta^-) = W - \sum_{v \in V \setminus \{\ell, r\}} \omega_v(\vartheta^-),$$

we get a simple set of linear inequalities, where each feasible solution provides a final weighting.

For the sake of completeness, such a solution can be found efficiently by replacing

$$\omega_\ell^* = W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \omega_r^*$$

in the inequality for ω_ℓ^* and combining

$$W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \min \{ \vartheta^- + \gamma, \omega_\ell^+(\vartheta^-) \} \leq \omega_r^* \leq W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \omega_\ell^-(\vartheta^-)$$

with the original inequality for ω_r^* . This results in the lower bound

$$\begin{aligned} & \max \{ \omega_r^-(\vartheta^-), W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \min \{ \vartheta^- + \gamma, \omega_\ell^+(\vartheta^-) \} \} \\ &= \max \{ \omega_r^-(\vartheta^-), W - \omega(V \setminus \{\ell, r\}, \vartheta^-) + \max \{ -\vartheta^- - \gamma, -\omega_\ell^+(\vartheta^-) \} \} \\ &= \max \{ \omega_r^-(\vartheta^-), W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \vartheta^- - \gamma, W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \omega_\ell^+(\vartheta^-) \} \end{aligned}$$

and the upper bound

$$\begin{aligned} & \min \{ \min \{ \vartheta^- + \gamma, \omega_r^+(\vartheta^-) \}, W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \omega_\ell^-(\vartheta^-) \} \\ &= \min \{ \vartheta^- + \gamma, \omega_r^+(\vartheta^-), W - \omega(V \setminus \{\ell, r\}, \vartheta^-) - \omega_\ell^-(\vartheta^-) \}, \end{aligned}$$

within which we can freely choose the value ω_r^* and derive the corresponding ω_ℓ^* from it.

Three Roots Without Attached Vertices

The path between ℓ and r might include more roots, over which we can distribute the remaining weight. We continue with the next slightly more difficult setting of three roots, where the additional root m shall be located in the middle of ℓ and r . Furthermore, we start with the assumption that no further vertices are attached to m .

The bounds for ℓ and r as derived in the beginning of this section are still valid. The two additional arcs rm and ℓm of A_R yield two additional weight bounds

$$\frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) - \omega(T_{\ell/r}, \vartheta) \leq \omega_m(\vartheta) \leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) - \omega(T_{\ell/r}, \vartheta),$$

similar to those of ℓ and r . Replacing the subtree weights of ℓ and r with their corresponding lower bounds $\omega^-(T_{\ell/r}, \vartheta)$ in the right relation provides

$$\omega_m(\vartheta) \leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) - \omega^-(T_{\ell/r}, \vartheta) = 2(\vartheta - \vartheta_{\frac{1}{2}}).$$

In case we have already $2(\vartheta^- - \vartheta_{\frac{1}{2}}) \leq \vartheta^- + \gamma$, being equivalent to $\vartheta^- \leq 2\vartheta_{\frac{1}{2}} + \gamma = \sigma(V) - W + \gamma$, for the current best lower bound ϑ^- , the weights can be chosen straightforwardly: Because the remaining weight is small enough, we can choose to set

$$\omega_m^* = 2(\vartheta^- - \vartheta_{\frac{1}{2}})$$

and use the lower bounds for the weights of the trees T_ℓ and T_r and thus the smallest possible weight for ℓ and r with

$$\omega_{\ell/r}^* = \omega_{\ell/r}^-(\vartheta^-).$$

We have

$$\omega^-(T_\ell, \vartheta^-) + \omega^-(T_r, \vartheta^-) + \omega_m^* = \frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) + \frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) + 2(\vartheta - \vartheta_{\frac{1}{2}}) = W$$

and ω_ℓ^* , ω_r^* and ω_m^* form together with $\omega_{V \setminus R}(\vartheta^-)$ an optimal weighting in $\Omega(\vartheta^-)$.

If we instead have $2(\vartheta^- - \vartheta_{\frac{1}{2}}) > \vartheta^- + \gamma$, equivalently $\vartheta^- > 2\vartheta_{\frac{1}{2}} + \gamma$, we need to distribute the weight in a different way. We have

$$\begin{aligned} W - \omega^{\leq}(T_{\ell}, \vartheta) - \omega^{\leq}(T_r, \vartheta) &\leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) - \omega^{\leq}(T_{\ell/r}, \vartheta) \\ \Leftrightarrow \omega^{\leq}(T_{r/\ell}, \vartheta) &\geq \frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) = \omega^-(T_{r/\ell}, \vartheta) \end{aligned}$$

and

$$\begin{aligned} W - \omega^{\leq}(T_{\ell}, \vartheta) - \omega^{\leq}(T_r, \vartheta) &\geq \frac{1}{2}W - (\vartheta - \vartheta_{\frac{1}{2}}) - \omega^{\leq}(T_{\ell/r}, \vartheta) \\ \Leftrightarrow \omega^{\leq}(T_{r/\ell}, \vartheta) &\leq \frac{1}{2}W + (\vartheta - \vartheta_{\frac{1}{2}}) = \omega^+(T_{r/\ell}, \vartheta). \end{aligned}$$

Therefore, setting

$$\omega_{\ell/r}(\vartheta) := \min \{ \vartheta + \gamma, \omega_{\ell/r}^+(\vartheta) \}$$

results in $\omega(T_{\ell/r}, \vartheta) = \omega^{\leq}(T_{\ell/r}, \vartheta)$ and, with

$$\omega_m(\vartheta) := W - \omega^{\leq}(T_{\ell}, \vartheta) - \omega^{\leq}(T_r, \vartheta),$$

the above bounds for the arcs rm and ℓm hold. However, at the same time, we need to ensure that the remaining weight in $\omega_m(\vartheta)$ does not exceed $\vartheta + \gamma$. By solving

$$W - \omega^{\leq}(T_{\ell}, \vartheta) - \omega^{\leq}(T_r, \vartheta) = \vartheta + \gamma$$

analogously to before, we get another lower bound ϑ_m^- on the optimal ϑ , to be maintained in ϑ^- . Again we have found the optimal weights with $\omega(\vartheta^-) \in \Omega(\vartheta^-)$.

More Than Three Roots or Attached Vertices

Finally, we consider the case where we have additional vertices in the form of further roots or of vertices that are attached to roots in the inner of the path formed by A_R , thus not to ℓ and r . Surprisingly, this constellation yields a straightforward solution in contrast to previous cases. Due to the distribution of the σ -values causing multiple roots as explained in Section 4.4.5, all attached vertices $v \in V \setminus (T_{\ell} \cup T_r \cup R)$ have $\sigma_v = 0$. Hence, they are outer vertices, whose weights are bounded by

$$\sigma(T_v) - \vartheta = -\vartheta \leq \omega(T_v, \vartheta) \leq \vartheta - \sigma(T_v) = \vartheta.$$

It would therefore be feasible to choose $\omega(T_v, \vartheta) = \vartheta - \vartheta_{\frac{1}{2}}$, for instance.

As we have seen before, $\omega(T_m, \vartheta) = \vartheta - \vartheta_{\frac{1}{2}}$ can also be a feasible choice for an additional root $m \in R \setminus \{\ell, r\}$. This means, if at least two such vertices exist, root or attached, we can proceed analogously to the beginning of the last section and set

$$\omega_{\ell/r}(\vartheta) := \omega_{\ell/r}^-(\vartheta).$$

The thus remaining weight of $2(\vartheta - \vartheta_{\frac{1}{2}})$ can be distributed with

$$\omega_{v/w}(\vartheta) := \vartheta - \vartheta_{\frac{1}{2}}$$

for two arbitrarily chosen vertices $v \neq w \in V \setminus (T_{\ell} \cup T_r)$. The remaining roots and attached vertices $V \setminus (T_{\ell} \cup T_r \cup \{v, w\})$ get a weight of 0. It is easy to check that this choice fulfils the inequalities corresponding to the arcs in A_R and all weights sum up to W . Hence, we have again found an optimal weighting with the constructed weight functions. As we do not introduce further lower bounds on ϑ this way, our formerly found current best lower bound ϑ^- is the optimal choice again to be inserted in the weight functions.

4.7 Integer Programming

Finally, we briefly investigate the consequences when restricting ourselves to integer parameters as well as integer variables in the previously presented problems. Surprisingly, with only slight adjustments, we can transfer the results for the continuous versions. In particular, we observe no increase in the runtime for the weight calculations in the different cases.

We still assume the graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, the strengths $\sigma \in \mathbb{N}^V$, the total weight $\mathbf{W} \in \mathbb{N}$ and the gap $\gamma \in \mathbb{N}_+$ to be given and fixed and consider the case $0 \leq W < \sigma(V)$. Note that \mathbb{N} includes 0 here, while we explicitly exclude 0 with \mathbb{N}_+ . We focus on the gapless version here, however with an integer gap, all results are easily extendable to the gapped versions. Furthermore, according to the experiences with the continuous versions, we mainly deal with trees.

4.7.1 The Problems

By replacing all appearances of \mathbb{R} with \mathbb{Z} , respectively, \mathbb{N} in the ZERO WEIGHT DISTRIBUTION PROBLEM and the STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES, we get the corresponding integer versions, which we add here for completeness:

STRENGTH-ONLY [FULL] INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES.

Given a tree $G = (V, E)$, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$ [and $\gamma \in \mathbb{N}_+$], find ϑ and ω that solve

$$\begin{aligned} & \min \vartheta \\ & \text{s.t. } \vartheta \in \mathbb{Z}, \omega \in \mathbb{Z}^V, \\ & \quad \vartheta \geq \sigma(T_a) + |\omega(T_a)| \quad \forall a \in A_{\text{out}}, \\ & \quad \vartheta \geq \vartheta_{\frac{1}{2}} + |\vartheta_{\frac{1}{2}} - \sigma(T_a) + \omega(T_a)| \quad \forall a \in A_{\text{in}}, \\ & \quad \omega(V) = W, \\ & \quad [\vartheta \geq \|\omega\|_{\infty} - \gamma]. \end{aligned}$$

Analogously, we get from Corollary 4.27 also

INTEGER ZERO WEIGHT DISTRIBUTION PROBLEM ON TREES. Given tree $G = (V, E)$ and $\sigma \in \mathbb{N}^V$, find ϑ that solves

$$\begin{aligned} & \min \vartheta \\ & \text{s.t. } \vartheta \in \mathbb{Z}, \\ & \quad \vartheta \geq \sigma(T_a) \quad \forall a \in A. \end{aligned}$$

When we consider the continuous versions with only integer parameters given, they form a relaxation of the above problems and therefore only provide lower bounds on the optimal value on the first sight. In the following, we however show that these bounds are tight in several cases. For this we work through the results of the previous sections and check if they still hold in the integer setting. We start the straightforward results where the relaxation directly yields optimal integer solutions.

4.7.2 Straightforward Insights

Zero Weight

First of all, we consider the zero weight problem handled in Section 4.3.2. As the restriction to fundamental cuts still holds, we can easily see that we have in the case of trees:

Corollary 4.74. *For G being a tree and $\sigma \in \mathbb{N}^V$, we have $\vartheta_0 \in \mathbb{N}$ for the optimal value of the ZERO WEIGHT DISTRIBUTION PROBLEM. Thus, $(\vartheta_0, \mathbb{O})$ is an optimal solution of the INTEGER ZERO WEIGHT DISTRIBUTION PROBLEM ON TREES.*

Proof. Clear by Theorem 4.16 with $\vartheta_0(e) \in \mathbb{N}$ for all $e \in E(G)$ for $\sigma \in \mathbb{N}^V$. \square

As the restriction to integer values does not increase the computational effort, we can also directly transfer Corollary 4.17:

Corollary 4.75. *If G is a tree, we can find an optimal solution for the INTEGER ZERO WEIGHT DISTRIBUTION PROBLEM ON TREES in $\mathcal{O}(|V|)$.*

Remember, due to the equivalence of the problems for $W = 0$, both above results also hold for the strength-only and the full problem in this case and we have

Corollary 4.76. *For $W = 0$, every optimal solution ϑ_0 to the INTEGER ZERO WEIGHT DISTRIBUTION PROBLEM ON TREES yields an optimal solution $(\vartheta_0, \mathbb{O})$ to the STRENGTH-ONLY and the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES and it can be found in $\mathcal{O}(|V|)$.*

Strength-only Problem

In case we do not have $W = 0$, we continue with the strength-only problem version as investigated in Section 4.5. With $\vartheta_0 \in \mathbb{N}$ and $\vartheta_{\frac{1}{2}} = \frac{1}{2}(\sigma(V) - W)$, we can also easily see the following claim:

Corollary 4.77. *For G being a tree, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$ and $\vartheta_W = \vartheta_0$ or $\vartheta_W = \vartheta_{\frac{1}{2}}$ with $\sigma(V)$ and W being either both even or both odd, we have $\vartheta_W \in \mathbb{N}$ for the optimal value of the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES.*

Proof. In case $\sigma(V)$ and W are either both even or both odd, we also have $\vartheta_{\frac{1}{2}} \in \mathbb{N}$ and the claim thus follows directly from Theorem 4.37, respectively, Lemma 4.55. \square

Note that this result holds whether or not the arc set A forms an arborescence in the tree G . However, it does not necessarily mean that we have found the objective value for integer problem version. It might be that we cannot find suitable integer weights fulfilling the defining inequalities, which means some $\omega^* \in \Omega(\vartheta_W) \cap \mathbb{Z}^V$, where we keep using the definition of $\Omega(\vartheta_W)$ over continuous weights specified in Lemma 4.28. However, with $\sigma(V) \in \mathbb{N}$ and $W \in \mathbb{N}$, we have $2\vartheta_{\frac{1}{2}} \in \mathbb{N}$ and the bounds defining $\Omega(\vartheta)$ of Definition 4.29 fulfil $\omega^+(T_v, \vartheta), \omega^-(T_v, \vartheta) \in \mathbb{Z}$ for all $v \in V \setminus \{r\}$ and all $\vartheta \in \mathbb{N}$.

That ϑ_W is indeed the optimal value of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES in the above cases can be seen with the simple formulas for the weight calculation derived in Section 4.5.2:

Corollary 4.78. *For G being a tree, A forming an arborescence, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$, $\vartheta_W \in \mathbb{N}$ and ω^* of Theorem 4.40, we have $\omega^* \in \Omega(\vartheta_W) \cap \mathbb{Z}^V$. Thus, (ϑ_W, ω^*) is an optimal solution of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES and can be computed in $\mathcal{O}(|V|)$.*

Proof. The construction of Theorem 4.40 still holds and provides integer weights with $\vartheta_W \in \mathbb{N}$ because only integer values are added or subtracted. The runtime is not influenced and thus remains as stated in Corollary 4.41. \square

We can easily see that this result is extendable to the improved weights of Lemma 4.42 established in Section 4.5.3, which are integer for integer input parameters analogously, and their non-negative counterparts of Theorem 4.45. As the deductions on the optimality still hold, too, we can further deduce analogously to Theorem 4.54:

Corollary 4.79. *For G being a tree, A forming an arborescence, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$, $\vartheta_W \in \mathbb{N}$ and ω^* of Theorem 4.45, we have $\omega^* \in \mathbb{N}^V$ and $\omega^* \in \Omega^*(\vartheta_W)$ or $\|\omega^*\|_\infty \leq \vartheta_W$.*

Even if we need to de-contract the problem instance in case the original tree does not have a unique root as described in Section 4.5.4, we can derive analogous results. In this case, we have $\vartheta_W = \vartheta_{\frac{1}{2}}$ according to Lemma 4.55. With $\sigma(T_a) = \frac{1}{2}\sigma(V) \in \mathbb{N}$ for $a \in A_R \neq \emptyset$, we know that $\sigma(V)$ is even in this case. Thus, for $\vartheta_{\frac{1}{2}}$ being an integer, W must be even, too. This means that the collapse of the inequalities for the arcs $a \in A_R$ to

$$\omega(T_a) = \sigma(T_a) - \vartheta_{\frac{1}{2}} = \frac{1}{2}\sigma(V) - \frac{1}{2}(\sigma(V) - W) = \frac{1}{2}W$$

leads to integer weights in the construction of Lemma 4.55. By replacing ω^* in the last two corollaries with the weights of Theorem 4.58 constructed from those of Lemma 4.55, we can therefore state the equivalent results for the uncontracted instance.

Corollary 4.80. *For G being a tree, A not forming an arborescence, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$, $\vartheta_W \in \mathbb{N}$ and ω^* of Theorem 4.58, we have $\omega^* \in \Omega(\vartheta_W) \cap \mathbb{Z}^V$. Thus, (ϑ_W, ω^*) is an optimal solution of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES and can be computed in $\mathcal{O}(|V|)$.*

Corollary 4.81. *For G being a tree, A not forming an arborescence, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$, $\vartheta_W \in \mathbb{N}$ and ω^* of Theorem 4.58, we have $\omega^* \in \mathbb{N}^V$ and $\omega^* \in \Omega^*(\vartheta_W)$ or $\|\omega^*\|_\infty \leq \vartheta_W$.*

This means, in case of $\vartheta_W \in \mathbb{N}$, we have solved the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES for arbitrary tree structures and we either have also solved the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES or need to apply different methods to tackle the full problem version, analogously to the continuous cases. First, we however investigate the case where $\vartheta_W \notin \mathbb{N}$.

4.7.3 Rounded Up

All of the previously handled cases exclude an optimal value of $\vartheta_{\frac{1}{2}} \notin \mathbb{N}$ for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES. As the continuous problem provides a lower bound on the integer version, a candidate for the optimal value is $\lceil \vartheta_{\frac{1}{2}} \rceil$. We can indeed confirm that $\lceil \vartheta_{\frac{1}{2}} \rceil$ holds the optimal value of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES.

Strength-only Problem

As $\vartheta_{\frac{1}{2}}$ is derived from the integer input parameters with $\vartheta_{\frac{1}{2}} = \frac{1}{2}(\sigma(V) - W)$, we can only have $\vartheta_{\frac{1}{2}} \notin \mathbb{N}$ if $\sigma(V) - W$ is odd, which means that one of the values $\sigma(V)$ and W is odd and the other is even. We further know that $2\vartheta_{\frac{1}{2}} \in \mathbb{N}$ and $\lceil \vartheta_{\frac{1}{2}} \rceil = \vartheta_{\frac{1}{2}} + 0.5$. Remember, we only have an optimal value of $\vartheta_{\frac{1}{2}}$ for the STRENGTH-ONLY WEIGHT DISTRIBUTION PROBLEM ON TREES in case of $A_{\text{in}} \neq \emptyset$.

By Corollary 4.30, we also have $\Omega(\vartheta_{\frac{1}{2}}) \subseteq \Omega(\lceil \vartheta_{\frac{1}{2}} \rceil)$. This means, if we had $\omega^* \in \mathbb{Z}^V$ for ω^* as constructed in Theorem 4.40 or in Lemma 4.42, we could directly deduce that $(\lceil \vartheta_{\frac{1}{2}} \rceil, \omega^*)$ is an optimal solution of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES. However, we see that we do in general not get integer weights by the constructions of both lemmata if we have $\vartheta_{\frac{1}{2}} \notin \mathbb{N}$. Therefore, we need to investigate the case in more detail.

By slightly modifying the weight construction of Lemma 4.42, we can see that we can easily construct integer weights:

Lemma 4.82. *For G being a tree, A forming an arborescence with root r and $\omega^* \in \mathbb{R}^V$ with*

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V_{\text{out}}, p_v \notin V_{\text{in}} \cup \{r\}, \\ \omega_v^* &= \lceil \vartheta_W \rceil - \sigma(T_v) & \forall v \in V_{\text{out}}, p_v \in V_{\text{in}} \cup \{r\}, \\ \omega_v^* &= \sigma(T_v) - \lfloor \vartheta_W \rfloor - \sum_{s \in S_{\text{in}}(v)} (\sigma(T_s) - \lfloor \vartheta_W \rfloor) - \sum_{s \in S_{\text{out}}(v)} (\lceil \vartheta_W \rceil - \sigma(T_s)) & \forall v \in V_{\text{in}}, \\ \omega_r^* &= W - \sum_{s \in S_{\text{in}}(r)} (\sigma(T_s) - \lfloor \vartheta_W \rfloor) - \sum_{s \in S_{\text{out}}(r)} (\lceil \vartheta_W \rceil - \sigma(T_s)), \end{aligned}$$

we have $\omega^* \in \Omega(\lceil \vartheta_W \rceil) \cap \mathbb{Z}^V$.

Proof. For the case $\vartheta_W \in \mathbb{N}$, we have $\lceil \vartheta_W \rceil = \lfloor \vartheta_W \rfloor = \vartheta_W$ and the weights are the same as in Lemma 4.42. For $\vartheta_W = \vartheta_{\frac{1}{2}} \notin \mathbb{N}$, apart from setting the subtree weight to a different upper bound with

$$\begin{aligned} \omega^*(T_v) &= \omega^+(T_v, \lceil \vartheta_{\frac{1}{2}} \rceil) = \lceil \vartheta_{\frac{1}{2}} \rceil + \sigma(T_v) - 2\vartheta_{\frac{1}{2}} \\ &= \sigma(T_v) - \vartheta_{\frac{1}{2}} + 0.5 \\ &= \sigma(T_v) - \lfloor \vartheta_{\frac{1}{2}} \rfloor \end{aligned}$$

for $v \in V_{\text{in}}$, the proof follows the one of Lemma 4.42 and thus in turn Theorem 4.40. \square

With these weights, we can now extend Corollary 4.78 with

Corollary 4.83. *For G being a tree, A forming an arborescence, $\sigma \in \mathbb{N}^V$, $W \in \mathbb{N}$ and ω^* of Lemma 4.82, $(\lceil \vartheta_W \rceil, \omega^*)$ is an optimal solution of the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES and can be computed in $\mathcal{O}(|V|)$.*

Again, we can also apply Theorem 4.45 to obtain the corresponding non-negative weights. Unfortunately, there is no collapse of the weight bounds and therefore of the inequalities defined by V_{in} anymore. Instead, we have a difference of 1 with

$$\omega^+(T_v, \lceil \vartheta_{\frac{1}{2}} \rceil) = \sigma(T_v) - \lfloor \vartheta_{\frac{1}{2}} \rfloor + 1 = \omega^-(T_v, \lfloor \vartheta_{\frac{1}{2}} \rfloor) + 1.$$

Thus, the arguments in the proof of Lemma 4.49 do not apply anymore and we can in general not establish the second-level optimality of these weights in the sense of Theorem 4.54 analogously to Corollary 4.79. However, by comparing the explicit formulas, we can easily see the following relation:

Corollary 4.84. *For $\tilde{\omega}^*$ of Lemma 4.42 and ω^* of Lemma 4.82, we have $\omega^* \leq \tilde{\omega}^* + \frac{1}{2}\mathbb{1}$.*

Thus, we can at least deduce that, if we already have a feasible solution for the FULL WEIGHT DISTRIBUTION PROBLEM ON TREES, the construction of the second-level optimal weights can be applied and provides a feasible solution for the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES:

Corollary 4.85. *For G being a tree, A forming an arborescence, $\vartheta_W = \vartheta_{\frac{1}{2}} \notin \mathbb{N}$, $\tilde{\omega}^*$ of Lemma 4.42 modified by Theorem 4.45 with $\|\tilde{\omega}^*\|_\infty \leq \vartheta_{\frac{1}{2}}$ and ω^* of Lemma 4.82 equivalently modified by Theorem 4.45, we have $\|\omega^*\|_\infty \leq \lceil \vartheta_{\frac{1}{2}} \rceil$.*

Proof. Clear with

$$\|\omega^*\|_\infty = \|\tilde{\omega}^* + \frac{1}{2}\mathbb{1}\|_\infty \leq \|\tilde{\omega}^*\| + \|\frac{1}{2}\mathbb{1}\|_\infty = \vartheta_{\frac{1}{2}} + \frac{1}{2} = \lceil \vartheta_{\frac{1}{2}} \rceil$$

because the modifications of Theorem 4.45 preserve the property of Corollary 4.84. \square

In case of an instance with multiple roots, we can again start with obtaining the weights for the contracted instance, which yields an integer problem as well if the uncontracted does. By slightly modifying the weights given in Lemma 4.55, we can also deal with the case where W is odd and $\frac{1}{2}W$ thus is not an integer, preventing the straightforward applicability of Lemma 4.55:

Corollary 4.86. *Every optimal solution $(\tilde{\omega}^*, \tilde{\omega}^*)$ to the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES over the contracted instance with, w.l.o.g.,*

$$\tilde{\omega}^*(T_\ell \setminus \{\ell\}) \geq \tilde{\omega}^*(T_r \setminus \{r\})$$

yields an optimal solution $(\lceil \vartheta_{\frac{1}{2}} \rceil, \omega^)$ to the STRENGTH-ONLY INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES over the uncontracted instance with*

$$\begin{aligned} \omega_v^* &= 0 & \forall v \in V \setminus (T_\ell \cup T_r), \\ \omega_v^* &= \tilde{\omega}_v^* & \forall v \in T_\ell \cup T_r \setminus \{\ell, r\}, \\ \omega_\ell^* &= \lfloor \frac{1}{2}W \rfloor - \tilde{\omega}^*(T_\ell \setminus \{\ell\}), \\ \omega_r^* &= \lfloor \frac{1}{2}W \rfloor - \tilde{\omega}^*(T_r \setminus \{r\}). \end{aligned}$$

Proof. The proof mainly follows the one of Lemma 4.55. However, instead of the collapse of the inequalities for $a \in A_R$, we obtain

$$\lfloor \frac{1}{2}W \rfloor \leq \omega(T_a) \leq \lceil \frac{1}{2}W \rceil$$

and the above weights thus are feasible. With $\tilde{\omega}^* \in \mathbb{N}$, we also have $\omega^* \in \mathbb{N}$. \square

This means that Corollaries 4.84 and 4.85 analogously apply for the case of multiple roots with the corresponding weights.

If the weights constructed for the different cases further fulfil $\max_{v \in V} \omega_v^* \leq \lceil \vartheta_{\frac{1}{2}} \rceil + \gamma$, we have again also found the optimal solution to the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES and in both cases preserved the linear runtime. For the remaining cases, we need to refer to the analogous strategies as introduced in Section 4.6, which we explain in more detail in the next section. There we however see that our algorithm is also suitable to produce integer weights.

Full Problem

If we have not yet found an optimal solution to the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES, which means we have $\|\omega^*\|_\infty > \lceil \vartheta_w \rceil + \gamma$ for the second-level optimal weights of the previous section, we need to proceed analogously to Section 4.6. There we have mainly dealt with two cases distinguishing between $A_{\text{in}} = \emptyset$ and $A_{\text{in}} \neq \emptyset$.

In case of $A_{\text{in}} = \emptyset$, the optimal value for the continuous problem has been given with $\vartheta_0 + \varepsilon^*$ for a certain ε^* in Section 4.6.1. Analogously to before, we can show that this lower bound on the integer problem is tight and we can simply round up to obtain the optimal value. Since we have $\vartheta_0 \in \mathbb{N}$ for integer input parameters, the rounding only concerns the found ε^* . Remember, in case of $A_{\text{in}} = \emptyset$, we always have a unique root r .

Lemma 4.87. *If G is a tree and we have $A_{\text{in}} = \emptyset$ and $\|\omega^*\|_\infty > \vartheta_0 + \gamma$ for ω^* as constructed in Theorem 4.45, the optimal solution to the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES is $(\vartheta_0 + \lceil \varepsilon^* \rceil, \omega(\lceil \varepsilon^* \rceil))$, with ε^* as given in Theorem 4.63 and the weight functions $\omega(\varepsilon)$ given in the proof of Theorem 4.63, and can be computed in $\mathcal{O}(|V|)$.*

Proof. In the construction of the proof of Theorem 4.63, it is easy to see that $\omega(\varepsilon) \in \Omega(\vartheta_0 + \varepsilon)$ provides integer weights if ε is integer. With

$$\begin{aligned} \omega_v(\lceil \varepsilon^* \rceil) &= \lceil \omega_v(\varepsilon^*) \rceil \quad \forall v \in S(r), \\ \omega_r(\lceil \varepsilon^* \rceil) &\leq \omega_r(\varepsilon^*), \end{aligned}$$

we further have

$$\|\omega(\lceil \varepsilon^* \rceil)\|_\infty \leq \vartheta_0 + \lceil \varepsilon^* \rceil + \gamma$$

and we have thus found a feasible integer solution holding the lower bound given by the relaxation, the continuous problem. The runtime remains as stated in Corollary 4.64. \square

In the same way, we can argue for the weight obtained by our algorithm for the remaining case $A_{\text{in}} \neq \emptyset$, which we have presented in Section 4.6.3:

Theorem 4.88. *The optimal value to the FULL INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES is $\lceil \vartheta_\diamond \rceil$ and we can find an optimal solution in $\mathcal{O}(|V|^3)$.*

Proof. The weight functions constructed in Section 4.6.3 provide integer solutions when inserting an integer value for ϑ . Those are also feasible because, by construction, we have $\omega(\vartheta) \in \Omega(\vartheta)$ and $\omega_v(\vartheta) \leq \vartheta + \gamma$ and thus also $\max_{v \in V} \omega_v(\lceil \vartheta_\diamond \rceil) \leq \lceil \vartheta_\diamond \rceil + \gamma$. According to the result for the relaxation given in Theorem 4.72, we have found the optimal integer solution. Furthermore, the runtime does not change compared to the one of the continuous problem given in Lemma 4.73. \square

In case of a tree where the arcs do not form an arborescence, we are confident that the strategies proposed in Section 4.6.4 are analogously applicable to obtain a de-contracted integer solution for the uncontracted instance. The constructed weight functions for the vertices apart from the roots yield integer values when applied to $\vartheta \in \mathbb{N}$ as well as the bounds for the balancing of the roots' weights. However, we do not show the details here because this goes beyond the scope of this work.

To put it in a nutshell, even the transfer to the integer programming setting does not increase the computational effort to solve the examined problems and the derived solution strategies for the continuous problem versions apply with only slight adjustments up to rounding.

5 Conclusions

Whether the quantum annealers, in particular those built by D-Wave, show an advantage over classical computers is still under discussion and will only reveal with the further development of such machines. To evaluate the great potential of this technology however, we need to design our experiments carefully. If the architecture does not change drastically, the two programming steps, minor embedding and parameter setting, will remain relevant in the long term. They are critical when it comes to providing meaningful input for the annealing machines. While the necessary minor embedding can prevent calculations on the machine at all, that is, if no embedding can be found, the specific parameter setting decisively influences the success in solving the actual problem.

First of all, this thesis gives a deep insight in the embedding problem within the setting of quantum annealing. With our hardness result of Section 3.2, the embedding of arbitrary graphs in broken Chimera or Pegasus graphs reveals to be as hard as the Ising problem, of which the machine shall actually solve a restricted version. However, the given proof is based on a very restrictive Chimera graph construction, whereas currently operating annealers have a very small ratio of broken vertices compared to the overall number of vertices. Although we could certainly decrease the number of broken vertices in our construction, some of them are indispensable, such as those in the odd vertex representation. Thus, it remains an open question whether a bounded ratio of broken vertices, for instance by a linear function in the size of the Chimera, still holds the same result.

Nevertheless, with a view to an increasing size of quantum chips, we need to apply different strategies to provide embeddings prior to the actual calculations. Taking up the idea of pre-calculated and reusable templates, our approach of Section 3.3 to handle the embedding problem for complete graphs in broken Chimera graphs demonstrates a possible solution. Considering operational times of the machines of over one year, the embeddings can be produced with a reasonable computational effort. With the help of dedicated bounding techniques, the branching strategy presented in Section 3.3.4 could be extended to a customized algorithm, which might show even better performance. In our future research, we also aim to transfer the construction of the model for the Chimera graph to the Pegasus graph, which is implemented in the latest quantum annealing architecture of D-Wave. The strong relation between both graphs is also evidenced by the transfer of the hardness result for the embedding problem. Due to the larger connectivity of the Pegasus graph, we even expect to obtain a model with less constraints caused by broken vertex pairs and thus a better solving performance compared to the Chimera.

The second programming step, the parameter setting, in this work also referred to as weight distribution, is as important as the embedding, however less deeply investigated in current research. In Chapter 4, we provide the first algorithms to find the coefficients of the embedded Ising model for a given model and its corresponding embedding such that both problems are provably equivalent, which means they yield equivalent solutions. Our results include a variety of cases, in particular arbitrary trees rather than only chains as well as the restriction to integer coefficients, and greatly improve the formerly known bounds. At the same time, they show that the parameter setting, in contrast to the embedding problem, is easy to solve, namely in

linear time for most cases or at least in cubic time for the general case. Furthermore, while the considered embedding problems are strongly related to the specific hardware graphs of D-Wave, our weight distribution algorithms are applicable for all hardware graphs which do not yield an all-to-all connectivity and thus require an embedding.

As the resulting problems can now directly be transferred to the D-Wave machines, the computational properties of these machines can now be investigated even further: Do the theoretically optimal coupling strengths also hold under the perturbations of the machine, that is, do they suffice to enforce the synchronization of the variables in practice? While, in theory, any positive gap is sufficient for the equivalence of the problems, an ideal quantum annealer should thus indeed return the optimal solution after a sufficiently long annealing time, the gap value most likely needs to be increased for any physical annealing machine. This gap parameter is a tool which provides different but, most importantly, equivalent encodings of the same problem and thus allows to study the difference between the theoretical and the *effective* coupling strength, which means the one which is necessary for the real machine to return the optimal value with an acceptable success probability. With this we will get a deeper understanding of the problem-independent behaviour of such machines, supporting their further development.

Lists

Problems

ISING PROBLEM	8
MINOR CONTAINMENT PROBLEM	14
EMBEDDING PROBLEM	14
MINOR EMBEDDING PROBLEM	15
BROKEN CHIMERA MINOR EMBEDDING PROBLEM	15
LARGEST COMPLETE GRAPH EMBEDDING PROBLEM	16
BROKEN CHIMERA RECOGNITION PROBLEM	18
BROKEN PEGASUS MINOR EMBEDDING PROBLEM	40
LARGEST COMPLETE GRAPH MATCHING PROBLEM	52
HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM	58
EMBEDDED ISING PROBLEM	64
[STRENGTHS-ONLY] GAPPED PARAMETER SETTING PROBLEM	74
STRENGTH-ONLY [FULL] GAPPED WEIGHT DISTRIBUTION PROBLEM	81
GAPPED ZERO WEIGHT DISTRIBUTION PROBLEM	81
STRENGTH-ONLY [FULL] WEIGHT DISTRIBUTION PROBLEM ON TREES	100
STRENGTH-ONLY [FULL] INTEGER WEIGHT DISTRIBUTION PROBLEM ON TREES . .	147
INTEGER ZERO WEIGHT DISTRIBUTION PROBLEM ON TREES	147

Figures

1.1 Layers of abstraction in transferring a problem to D-Wave's quantum annealer . .	3
2.1 Bipartition in the Chimera graph $C_{3,3}$	11
3.1 Standard triangular complete graph embedding in the ideal Chimera graph . . .	15
(a) complete graph	
(b) standard triangular embedding	
3.2 Polynomial reduction steps of the proof	19
3.3 Grid tentacle example	21
3.4 Example graph and its rectangular representation	22
(a) $B^* \in \mathcal{B}$	
(b) $R(B^*)$	
3.5 Manipulation of grid vertices and incident edges of the rectangular representation	23
(a) even vertex	
(b) odd vertex	
3.6 Enlarged rectangular representation $L(B^*)$	24
3.7 Broken Chimera graph representing a vertex	25

3.8	Chimera tentacles	26
	(a) unmodified version	
	(b) modified version	
3.9	Combined Chimera graph elements representing neighboring vertices	27
3.10	Hamiltonian cycles in the example graph and its rectangular representation	28
	(a) in B^*	
	(b) in $R(B^*)$	
3.11	Different possibilities for parts of a Hamiltonian path to cross a unit cell	29
3.12	Hamiltonian cycle construction in the Chimera tentacle example	30
	(a) illustration in the grid tentacle	
	(b) straight elements	
	(c) corner	
	(d) closing loop	
	(e) Chimera tentacle with a possible Hamiltonian cycle	
3.13	Hamiltonian path construction in a Chimera strip example	31
	(a) snake path from a vertical vertex to a vertical vertex	
	(b) illustration of (a) in the grid strip	
	(c) snake path from a vertical vertex to a horizontal vertex	
	(d) illustration of (c) in the grid strip	
3.14	Hamiltonian path construction in the Chimera tentacle example	33
	(a) illustration in the grid tentacle	
	(b) straight elements	
	(c) inner corner	
	(d) outer corner	
	(e) Chimera tentacle with a possible Hamiltonian path	
3.15	Vertices joining the vertex Chimera element with the edge Chimera elements	35
3.16	Parts of the Hamiltonian cycle illustrated in the underlying grid graph	36
	(a) even vertex with loop to the right	
	(b) odd vertex with loop from the right	
	(c) even vertex with loop to the left	
	(d) odd vertex with loop from the left	
	(e) even vertex with loop to below	
	(f) odd vertex with loop from below	
3.17	Parts of the Hamiltonian cycle through the Chimera vertex elements	37
	(a) even vertex with loop to the right	
	(b) odd vertex with loop from the right	
	(c) even vertex with loop to the left	
	(d) odd vertex with loop from the left	
	(e) even vertex with loop to below	
	(f) odd vertex with loop from below	
3.18	Parts of the Hamiltonian cycle in neighboring vertices	38
3.19	Cycle in $L(B^*)$ illustrating the Hamiltonian cycle in $C(B^*)$	38
3.20	Inner of the Pegasus graph P_5	42
3.21	Different variants of complete graph embeddings in the ideal Chimera graph	43
	(a) extended triangular	
	(b) permutation of diagonal elements	
3.22	Complete graph embeddings in a broken Chimera graph with permuted crossroads	44
	(a) permutation over all rows and columns	
	(b) crosses in broken Chimera	

(c) crossroads in broken Chimera to be found	
3.23 Specific indexing in the Chimera graph	46
(a) horizontal vertices	
(b) vertical vertices	
(c) edge examples	
3.24 Crosses that do not meet due to broken vertices	48
(a) in different unit cell rows	
(b) in the same unit cell row	
3.25 Sets of mutually exclusive crossroads caused by two broken vertical vertices	49
(a) two different MES due to different rows	
(b) MES in same row already handled by matching constraints	
3.26 Illustration of the interval function	50
3.27 Mutually exclusive crossroads caused by two different broken vertices	51
(a) four different relations of positions	
(b) single crossroad being pairwise forbidden with all crossroads in rectangle	
(c) no further constraints due to same unit cell row	
3.28 Solutions for a very broken Chimera graph	57
(a) by ILP	
(b) optimal	
3.29 Complete graph sizes against Chimera sizes	62
4.1 Example for an embedded subgraph structure	72
4.2 Illustration of case distinctions	86
4.3 Connected components for $S^* \in \mathcal{C}' \setminus \mathcal{C}$	95
4.4 Example tree with inner and outer arcs	99
4.5 Arborescence with inner and outer vertices	103
4.6 Illustration of the σ -distribution for the case of multiple root	104
4.7 Example of a contraction of a tree to obtain a unique root	105
4.8 Compression of the tree to obtain only inner arcs and vertices	110
4.9 Compression of the tree keeping the next outer vertices	113
4.10 Arborescence with only outer vertices	124
4.11 Illustration of the two different cases in the iteration	126
(a) case A	
(b) case B	
4.12 Illustration of different cases of $\omega_\ell(\vartheta)$ for some leaf $\ell \in V_{\text{in}}$	133
(a) with $ S_{\text{out}}(\ell) = 0$	
(b) with $ S_{\text{out}}(\ell) = 1$	
(c) with $ S_{\text{out}}(\ell) = 2$	
4.13 Illustration of $\omega_v(\vartheta)$ for $v \in V_{\text{in}}$	135
4.14 Illustration of the sum of two piecewise linear functions	136

Tables

3.1 Averaged solution ratios for each combination of size and ratio of broken vertices	61
(a) LARGEST COMPLETE GRAPH MATCHING PROBLEM	
(b) HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM with $m = 0.25$	
(c) HEURISTIC LARGEST COMPLETE GRAPH MATCHING PROBLEM with $m = 0.0$	
3.2 Difference of rounded averaged solution ratios from heuristic models	61

Algorithms

4.1	Get Optimal Strength	138
4.2	Helper Functions	139

Bibliography

- [1] I. Adler, F. Dorn, F. V. Fomin, I. Sau, and D. M. Thilikos. Fast minor testing in planar graphs. *Algorithmica*, 64(1):69–84, 2012. DOI: 10.1007/s00453-011-9563-9.
- [2] I. Adler, F. Dorn, F. V. Fomin, I. Sau, and D. M. Thilikos. Faster parameterized algorithms for minor containment. *Theoretical Computer Science*, 412(50):7018–7028, 2011. DOI: 10.1016/j.tcs.2011.09.015.
- [3] D. Aharonov, W. Van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008. DOI: 10.1137/080734479.
- [4] T. Albash and D. A. Lidar. Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002, 2018. DOI: 10.1103/RevModPhys.90.015002.
- [5] F. Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982. DOI: 10.1088/0305-4470/15/10/028.
- [6] K. Boothby, P. Bunyk, J. Raymond, and A. Roy. Next-generation topology of D-Wave quantum processors. *preprint*, 2020. arXiv: 2003.00133 [quant-ph].
- [7] T. Boothby, A. D. King, and A. Roy. Fast clique minor generation in chimera qubit connectivity graphs. *Quantum Information Processing*, 15(1):495–508, 2016. DOI: 10.1007/s11128-015-1150-6.
- [8] M. Born and V. Fock. Beweis des Adiabatenatzes [Proof of the adiabatic theorem]. German. *Zeitschrift für Physik*, 51(3-4):165–180, 1928. DOI: 10.1007/BF01343193.
- [9] E. Boros, P. Hammer, and G. Tavares. Preprocessing of quadratic unconstrained binary optimization. Technical report, Technical report RRR 10-2006, RUTCOR research report, 2006.
- [10] E. Boros and P. L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1):155–225, 2002. DOI: 10.1016/S0166-218X(01)00341-9.
- [11] J. Cai, W. G. Macready, and A. Roy. A practical heuristic for finding graph minors. *preprint*, 2014. arXiv: 1406.2741 [quant-ph].
- [12] V. Choi. Minor-embedding in adiabatic quantum computation: I. The parameter setting problem. *Quantum Information Processing*, 7(5):193–209, 2008. DOI: 10.1007/s11128-008-0082-9.
- [13] V. Choi. Minor-embedding in adiabatic quantum computation: II. Minor-universal graph design. *Quantum Information Processing*, 10(3):343–353, 2011. DOI: 10.1007/s11128-010-0200-3.
- [14] D-Wave Systems Inc. D-Wave System documentation. <https://docs.dwavesys.com/docs/latest/index.html>. visited 2022-01-21.
- [15] D-Wave Systems Inc. Dwave-system. *GitHub repository*, 2021. version 1.12.0 https://github.com/dwavesystems/dwave-system/blob/1.12.0/dwave/embedding/chain_strength.py.

- [16] D-Wave Systems Inc. Minorminer. *GitHub repository*, 2021. version 0.2.7. <https://github.com/dwavesystems/minorminer/releases/tag/0.2.7>.
- [17] A. Das and B. K. Chakrabarti. Colloquium: Quantum annealing and analog quantum computation. *Reviews of Modern Physics*, 80(3):1061–1081, Sept. 2008. DOI: 10.1103/revmodphys.80.1061.
- [18] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 5. Edition, 2017. DOI: 10.1007/978-3-662-53622-3.
- [19] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Quantum computation by adiabatic evolution. *preprint*, 2000. arXiv: quant-ph/0001106v1.
- [20] M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. Series of Books in the Mathematical Sciences. W. H. Freeman and Company, 1979.
- [21] A. Gleixner, M. Bastubbe, L. Eifler, T. Gally, G. Gamrath, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. E. Lübbecke, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, C. Schubert, F. Serrano, Y. Shinano, J. M. Viernickel, M. Walter, F. Wegscheider, J. T. Witt, and J. Witzig. The SCIP Optimization Suite 6.0. Technical Report, Optimization Online, July 2018. http://www.optimization-online.org/DB_FILE/2018/07/6692.pdf.
- [22] T. D. Goodrich, B. D. Sullivan, and T. S. Humble. Optimizing adiabatic quantum program compilation using a graph-theoretic framework. *Quantum Information Processing*, 17(5):118, 2018. DOI: 10.1007/s11128-018-1863-4.
- [23] K. E. Hamilton and T. S. Humble. Identifying the minor set cover of dense connected bipartite graphs via random matching edge sets. *Quantum Information Processing*, 16(4):94, 2017. DOI: 10.1007/s11128-016-1513-7.
- [24] I. V. Hicks. Branch decompositions and minor containment. *Networks: An International Journal*, 43(1):1–9, 2004. DOI: 10.1002/net.10099.
- [25] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006. DOI: 10.1090/S0273-0979-06-01126-8.
- [26] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. DOI: 10.1137/0202019.
- [27] E. Ising. Beitrag zur Theorie des Ferromagnetismus [contribution to the theory of ferromagnetism]. German. *Zeitschrift für Physik*, 31(1):253–258, 1925. DOI: 10.1007/BF02980577.
- [28] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982. DOI: 10.1137/0211056.
- [29] M. W. Johnson, M. H. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, et al. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 2011. DOI: 10.1038/nature10012.
- [30] M. Jünger, E. Lobe, P. Mutzel, G. Reinelt, F. Rendl, G. Rinaldi, and T. Stollenwerk. Quantum annealing versus digital computing: An experimental comparison. *Journal of Experimental Algorithmics*, 26:1–30, 2021. DOI: 10.1145/3459606.
- [31] A. D. King and C. C. McGeoch. Algorithm engineering for a quantum annealing platform, 2014. arXiv: 1410.2628 [cs.DS].

-
- [32] C. Klymko, B. D. Sullivan, and T. S. Humble. Adiabatic quantum programming: Minor embedding with hard faults. *Quantum Information Processing*, 13(3):709–729, 2014. DOI: 10.1007/s11128-013-0683-9.
- [33] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang. The unconstrained binary quadratic programming problem: A survey. *Journal of Combinatorial Optimization*, 28(1):58–81, 2014. DOI: 10.1007/s10878-014-9734-0.
- [34] B. Korte and J. Vygen. *Combinatorial optimization: Theory and algorithms*, volume 21 of *Algorithms and Combinatorics*. Springer, 6. Edition, 2018. DOI: 10.1007/978-3-662-56039-6.
- [35] E. Lobe. *Quadratische binäre Optimierung ohne Nebenbedingungen auf Chimera-Graphen [Quadratic binary optimization without constraints on Chimera graphs]*. German. Master’s thesis, Otto-von-Guericke-Universität Magdeburg, 2016. <https://elib.dlr.de/112063/>.
- [36] E. Lobe and A. Lutz. Minor embedding in broken Chimera and Pegasus graphs is NP-complete. *preprint*, 2021. arXiv: 2110.08325 [quant-ph].
- [37] E. Lobe, L. Schürmann, and T. Stollenwerk. Embedding of complete graphs in broken Chimera graphs. *Quantum Information Processing*, 20(7):1–27, 2021. DOI: 10.1007/s11128-021-03168-z.
- [38] A. Lucas. Ising formulations of many NP problems. *Frontiers in physics*, 2:5, 2014. DOI: 10.3389/fphy.2014.00005.
- [39] S. Maher, M. Miltenberger, J. P. Pedroso, D. Rehfeldt, R. Schwarz, and F. Serrano. PySCIPOpt: Mathematical programming in Python with the SCIP optimization suite. In *Mathematical Software – ICMS 2016*, pages 301–307. Springer International Publishing, 2016. DOI: 10.1007/978-3-319-42432-3_37.
- [40] D. W. Matula and F. Shahrokhi. Sparsest cuts and bottlenecks in graphs. *Discrete Applied Mathematics*, 27(1-2):113–123, 1990. DOI: 10.1016/0166-218X(90)90133-W.
- [41] C. C. McGeoch. Theory versus practice in annealing-based quantum computing. *Theoretical Computer Science*, 816:169–183, 2020. DOI: 10.1016/j.tcs.2020.01.024.
- [42] H. Neven, V. S. Denchev, M. Drew-Brook, J. Zhang, W. G. Macready, and G. Rose. Nips 2009 demonstration: Binary classification using hardware implementation of quantum annealing. *Quantum*, 2009.
- [43] J. P. Pinilla and S. J. Wilton. Layout-aware embedding for quantum annealing processors. In *International Conference on High Performance Computing*, pages 121–139, Berlin. Springer, 2019. DOI: 10.1007/978-3-030-20656-7_7.
- [44] J. Raymond, N. Ndiaye, G. Rayaprolu, and A. D. King. Improving performance of logical qubits by parameter tuning and topology compensation. In *IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 295–305, Oct. 2020. DOI: 10.1109/qce49297.2020.00044.
- [45] E. G. Rieffel, D. Venturelli, B. O’Gorman, M. B. Do, E. M. Prystay, and V. N. Smelyanskiy. A case study in programming a quantum annealer for hard operational planning problems. *Quantum Information Processing*, 14(1):1–36, 2015. DOI: 10.1007/s11128-014-0892-x.
- [46] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995. DOI: 10.1006/jctb.1995.1006.
- [47] T. Serra, T. Huang, A. Raghunathan, and D. Bergman. Template-based minor embedding for adiabatic quantum optimization. *preprint*, 2019. arXiv: 1910.02179 [cs.DS].

- [48] T. Stollenwerk, E. Lobe, and M. Jung. Flight gate assignment with a quantum annealer. In *International Workshop on Quantum Technology and Optimization Problems*, pages 99–110, Berlin. Springer, 2019. DOI: 10.1007/978-3-030-14082-3_9.
- [49] T. Stollenwerk, B. O’Gorman, D. Venturelli, S. Mandra, O. Rodionova, H. Ng, B. Sridhar, E. G. Rieffel, and R. Biswas. Quantum annealing applied to de-conflicting optimal trajectories for air traffic management. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):285–297, 2019. DOI: 10.1109/TITS.2019.2891235.
- [50] S. L. Tanimoto, A. Itai, and M. Rodeh. Some matching problems for bipartite graphs. *Journal of the ACM*, 25(4):517–525, 1978. DOI: 10.1145/322092.322093.
- [51] D. Venturelli, D. J. Marchand, and G. Rojo. Quantum annealing implementation of job-shop scheduling. *preprint*, 2015. arXiv: 1506.08479 [quant-ph].
- [52] A. Zaribafiyani, D. J. Marchand, and S. S. C. Rezaei. Systematic and deterministic graph minor embedding for cartesian products of graphs. *Quantum Information Processing*, 16(136), 2017. DOI: 10.1007/s11128-017-1569-z.
- [53] S. Zbinden, A. Bärtschi, H. Djidjev, and S. Eidenbenz. Embedding algorithms for quantum annealers with Chimera and Pegasus connection topologies. In *International Conference on High Performance Computing*, pages 187–206, Berlin. Springer, 2020. DOI: 10.1007/978-3-030-50743-5_10.