Designing and Expanding Electrical Networks

–

Complexity and Combinatorial Algorithms

Zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Matthias Wolf

Tag der mündlichen Prüfung:	13.2.2023
Erste Referentin:	Prof. Dr. Dorothea Wagner
Zweiter Referent:	Prof. Dr. Thomas Brown

Acknowledgement

First of all, I want to thank Dorothea Wagner for giving me the opportunity to join her group. I am particularly grateful that I was able to travel to conferences in many countries, which really broadened my horizon and allowed me to meet interesting people and visit fascinating places.

I would also like to thank all colleagues at the Institute of Theoretical Informatics. We had many (and sometimes not too serious) discussions during breaks. In particular, I want to thank my office mates Sascha Gritzbach and Franziska Wegner. We spent a lot of time together (in particular, close to deadlines). The (occasionally long) discussions about our research topics (and non-research related topics) we really helpful and kept me motivated.

Moreover, I thank Lilian Beckert, Isabelle Junge, Laurette Lauffer, and Tanja Wehrmann for helping me navigate the university bureaucracy, and Ralf Kölmel for his technical support.

Furthermore, I thank all members of the research training group "Energy Status Data", and in particular the members of my self-organized study group. They provided an interdisciplinary view on energy systems, and I learned a lot from them.

I thank Tom Brown, who readily agreed to referee this thesis. I also like to thank Max Geißer, Sascha Gritzbach, and my parents for proofreading parts of this thesis.

Finally, I thank my friends and family for providing me with support when needed.

Abstract

The transition from conventional to renewable power generation has a large impact on when and where electricity is generated. To deal with this change the electric transmission network needs to be adapted and expanded. Expanding the network has two benefits. Electricity can be generated at locations with high renewable energy potentials and then transmitted to the consumers via the transmission network. Without the expansion the existing transmission network may be unable to cope with the transmission needs, thus requiring power generation at locations closer to the energy demand, but at less well-suited locations. Second, renewable energy generation (e.g., from wind or solar irradiation) is typically volatile. Having strong interconnections between regions within a large geographical area allows to the smooth the generation and demand over that area. This smoothing makes them more predictable and the volatility of the generation easier to handle.

In this thesis we consider problems that arise when designing and expanding electric transmission networks. As the first step we formalize them such that we have a precise mathematical problem formulation. Afterwards, we pursue two goals: first, improve the theoretical understanding of these problems by determining their computational complexity under various restrictions, and second, develop algorithms that can solve these problems.

A basic formulation of the expansion planning problem models the network as a graph and potential new transmission lines as edges that may be added to the graph. We formalize this formulation as the problems Flow Expansion and Electrical Flow Expansion, which differ in the flow model (graph-theoretical vs. electrical flow). We prove that in general the decision variants of these problems are $\mathcal{NP}$-

complete, even if the network structure is already very simple, e.g., a star. For certain restrictions, we give polynomial-time algorithms as well. Our results delineate the boundary between the $\mathcal{NP}$-complete cases and the cases that can be solved in polynomial time.

The basic expansion planning problems mentioned above ignore that real transmission networks should still be able to operate if a small part of the transmission equipment fails. We employ a criticality measure from the literature, which measures the dynamic effects of the failure of a single transmission line on the whole transmission network. In a first step, we compare this criticality measure to the well-used $N-1$ criterion. Moreover, we formulate this criticality measure as a set of linear inequalities, which may be added to any formulation of a network design problem as a mathematical program. To exemplify this usage, we introduce the criticality criterion in two transmission network expansion planning problems, which can be formulated as mixed-integer linear programs (MILPs). We then evaluate the performance of solving the MILPs. Finally, we develop a greedy heuristic for one of the two problems, and compare its performance to solving the MILP.

Microgrids play an important role in the electrification of rural areas. We formalize the design of the cable layout of a microgrid as a geometric optimization problem, which we call Microgrid Cable Layout. A key difference to the network design problems above is that there is no graph with candidate edges given. Instead, edges and new vertices may be placed anywhere in the plane. We present a hybrid genetic algorithm for Microgrid Cable Layout and evaluate it on a set of benchmark instances, which include a real microgrid in the Democratic Republic of the Congo.

Finally, instead of expanding electrical networks one may place electric equipment such as FACTS (flexible AC transmission system). These influence the properties of the transmission lines such that the network can be used more efficiently. We apply a model of FACTS from the literature and study the problem whether a given network with given positions and properties of the FACTS admits an electrical flow provided that FACTS are set appropriately. We call such a flow a *FACTS flow*. In this thesis we prove that in general it is $\mathcal{NP}$-complete to determine whether a network admits a FACTS flow, and we present polynomial-time algorithms for two restricted cases.

Contents

1 Introduction

In the Paris Agreement the United Nations (UN) have pledged to keep "the global average temperature to well below 2 °C above pre-industrial levels" and to make efforts to limit the temperature increase to 1.5 °C above pre-industrial levels" [Uni15a]. To reach this goal a reduction of global greenhouse gas emissions is necessary. In order to still satisfy the global energy demand there needs to be a shift from fossil fuels to renewable energy sources. Increasing the share of these in the global energy mix until 2030 is also part of the UN's Sustainable Development Goals (SDGs) as SDG 7.2 [Uni15b]. A recent report on these goals concluded that the share of renewable energy in total final energy consumption increased slightly from 16.4 % in 2010 to 17.1 % in 2018 [Uni21, p. 41]. The main contributor of this increase was the electricity sector with an increase from 19.7 % to 25.4 % in the same time period. Similar data is reported in the latest REN21 global status report with an increase from 20.4 % in 2011 to 28.3 % in 2021.

In their European Green Deal the European Commission formulated the goal to achieve climate neutrality in the European Union (EU) by 2050 [Eur19]. According to the 2030 targets of the EU at least 32 % of energy consumed shall be obtained from renewable energy sources. As a reference, in 2020 the share of renewables in the gross final energy consumption was 22.1 % [Eur22, p. 142].

Even though electricity makes up only about 20 % of the global final energy consumption (21 % according to [Uni21], 17 % for the power sector according to [Ren22b]), it plays a crucial role in reducing greenhouse gas emissions. Machines powered commonly by fossil fuels are replaced by electric ones. For example in the transportation sector, battery electric vehicles slowly replace vehicles with internal combustion

engines. In the heating sector, one may install electric heat pumps instead of oil and gas heating. Therefore, it is expected that the share of electricity in the total energy consumption increases. Based on current and announced policies the International Energy Agency (IEA) estimates that the share of electricity among the useful energy consumption rises from 30 % in 2020 to 34 % in 2030 [Int21b, p. 197]. This relative increase corresponds to an absolute increase as well. They estimate that the global electricity demand grows from 23 300 TW h in 2020 to almost 30 000 TW h in 2030 and 42 000 TW h in 2050 [Int21b, p. 195].

Increasing the share of renewables in the electricity production presents challenges. Conventional fossil fuel based power plants have a much higher power capacity than wind turbines or photovoltaic power plants. Consequently, there need to be much more of these. Moreover, renewable power plants have different requirements on the environment in which they are built. For example, wind turbines are more effective in areas with constantly high wind speeds. This in particular leads to the commissioning of offshore wind parks.

But renewable power plants are not only more distributed spatially than conventional power plants, they also depend strongly on the local environment and weather. For example, a photovoltaic installation can only produce electricity when the sun is shining, and a wind turbine requires sufficiently high (but not too high) wind speed. These problems can be alleviated by selecting sites for the installations where the conditions allow for a higher and more constant energy yield, but this does not address the problems completely. The dependency of renewable generation on external factors such as weather means that they cannot be used whenever needed, but rather they must be used whenever the conditions allow. This means that they are *non-dispatchable*. The intermittent nature of renewables and the high variability in their power generation makes managing the electricity system much harder.

Moreover, to ensure constant access to electricity, one needs to deal with the fact that there may be times when renewable generation is insufficient to meet the current demand. In particular, this may happen when there is little wind and it is cloudy or at night. Then, neither wind turbines nor photovoltaic plants are able to generate much electric power. In particular in Germany, such a time period that lasts more than one day is known under the German term *Dunkelflaute*, which can be roughly translated as “dark wind lull” [LBWR21]. In countries surrounding the North Sea and the Baltic Sea such events occur mainly in November, December, and January, and on average 2 to 4 days in each of these months [LBWR21]. There are several, not mutually exclusive ways to address the challenges posed by the increased volatility of the electricity generation.

Storage. Increase the ability of the power grid to store energy in times of electricity surplus, which can later be fed into the grid in times of electricity deficit. There are numerous storage technologies available—all with different characteristics such as the storage duration, the amount of energy that can be stored, the peak power that can be delivered, and the costs of storage per unit of energy.

One form of energy storage uses basically the same technology as small electronic devices: batteries, and in fact, often even lithium-ion batteries. Examples of such batteries with a storage capacity of up to a few MW h integrated into wind farms can be found in [Win17]. But there are also larger storage facilities, for example the Moss Landing Energy Storage Facility in California with a total capacity of 400 MW and 1600 MW h [Vis21]. The owner claims this to be the world's largest lithium-ion battery storage system.

But surplus electric energy can also be stored in different energy forms. In fact, in the EU pumped hydro storage is the main form of energy storage [Eur+20]. It is also possible to store energy in the form of heat, e.g., using molten salt. Often, the medium is heated directly by solar irradiance in concentrated solar power plants, allowing the electricity production to be shifted into the night [BOB21]. But there is also a demonstration projects that uses surplus electricity to heat a medium (vulcanic rocks in this case), and later transforms the stored thermal energy again to electric energy [NS 20, Sie22]. In this project the maximum storage duration is given as seven days, and it may power up to 1500 households [NS 20].

Grid expansion. Increase the ability of the power grid to transmit power from far away regions by adding transmission capacities to the grid, typically by building new transmission lines. This allows the variability of the electricity generation to be smoothed over a larger area. The larger the area is, the less likely it is that all parts of the area are affected by the weather in the same way. For example, a study encompassing 11 countries bordering the North and Baltic Sea concluded that interconnecting the countries reduces the mean frequency of a Dunkelflaute from 3–9 % for the individual countries to 3.5 % for the whole region [LBWR21].

For example, in Germany the Bundesnetzagentur—Germany's regulatory body for, among others, electricity—mentions 101 expansion projects of the transmission grid [Bun22]. The total length of these projects is approximately 12 256 km, of which 2005 km have been completed. Notable projects among these are the long distance high-voltage direct current (DC) lines: *A-Nord*, *Ultranet*, *SuedLink*, and *SuedOstLink*. They run roughly north to south and connect the regions in Northern Germany with high wind power capacities to South Germany.

Similarly, in the United States of America (USA) there are several projects that aim to increase the transmission capacity between areas with large renewable generation

and more populated areas. For example, both the *Gateway West* [Pac21] and *TransWest Express* shall connect wind farms in Wyoming to consumers on the West Coast. Both include more than 1000 km of high-voltage transmission lines. A similar project is the *Grain Belt Express* connecting wind farms in Kansas to Indiana via more than 1250 km of transmission lines [Inv22].

Demand response. Shift the electricity load from times when renewable energy is scarce to times when it is abundant. In this way the demand follows the availability of electricity generation. This represents a change of the way people interact with the electricity grid away from the current assumption that the electricity generation is responsible for matching the electricity demand at any time.

A history of demand response (and the more general demand-side management), focused mostly on the USA, can be found in [Gel17]. One possibility for demand response is to exploit the flexibility offered by power-to-heat. For example, Gjorgievski et al. [GMAD21] list 34 projects of power-to-heat demand response. One of these aims to reduce the peak loads in the Australian power grid by managing residential air conditioners [Aus22]. A review of 16 field studies on the suitability and maturity of demand response from power-to-heat concludes that "significant benefits have been demonstrated" [Koh+19, p. 543].

A tighter coupling of the electricity grid with other sectors, in particular, the heating sector, allows to exploit the flexibilities there as well. In the paragraph on energy storage above it is mentioned that surplus electric energy may be used to heat materials such as sand and the heat is later converted to electric energy again [NS 20]. There are similar projects that use the stored heat directly in the form of district heating [Ren22a]. One may also consider the combination of the two uses such that the stored heat may either be transformed to electric energy or directly used based on what is more beneficial at the moment. Such a combined approach showcases the potential tighter sector coupling has for the future energy system.

It should be noted that the mentioned techniques should not be considered mutually exclusive. A cost-effective solution will likely use a combination of the techniques presented above. There are modeling tools that can help finding good solutions, e.g., [ZMT11, BHS18, Thu+18, DIg22, HOM22]. An overview over modeling tools for energy and electrical systems is given by Ringkjøb, Haugan, and Solbrekke [RHS18]. The exact results however strongly depend on the assumptions the planners make. But grid expansion is often part of a good solution [SSH12, Bro+18, CPHL21, Gea+21]. One study indicates that allowing for more additional transmission capacity reduces the overall system costs up to some optimal transmission capacity [Bro+18]. In another study the scenario with the lowest investment in transmission expansion

has the highest overall system costs, and in general it finds that their solutions invest more in transmission expansion than in storage [CPHL21]. This shows that optimal solutions to the models typically include grid expansion.

In practice, there are multiple examples where missing grid infrastructure slows the adoption of renewable energy sources. For example, in Germany offshore wind installations that were awarded in 2018 were postponed until the necessary grid infrastructure is built. It is expected that they are commissioned in 2022 or 2023 [Win21, p. 13]. Similarly, transmission and distribution grid expansion is required in Poland [Int21a, p. 65] and Latin America [Int21a, p. 71]. In general, the IEA views limits in the transmission and distribution infrastructure as one of the three key challenges to add more photovoltaic and wind power installations [Int21a, p. 39].

All these modeling results and real-world observations indicate that the expansion of the power grid will likely play a crucial role in decarbonizing the energy system. In this thesis we therefore focus on the computational problems that occur in the context of expanding electric transmission grids.

Literature on Transmission Network Expansion Planning

Expanding the electric transmission grid is a vast topic, which raises a lot of research questions. Coupled with the high relevance of the topic, it is not surprising that there is a huge body of literature on transmission network expansion planning. We give a brief overview over the research area, and include pointers to literature for further reading.

The book edited by Lumbreras, Abdi, and Ramos [LAR21] gives a good introduction to transmission network expansion planning problems. A basic formulation considers a given power grid and potential candidate lines that may be added to fulfill future demand. The objective is to minimize the investment costs, the operation costs, and the reliability costs, i.e., costs that occur to ensure the reliable operation of the power grid such as costs of load curtailment. There are multiple choices for the power flow model (transport model, which is a graph-theoretical flow; linearized alternating current (AC) power flow, often called DC power flow; and AC power flow). But for transmission network expansion planning the most common choice is the linearized AC power flow model [LARM21, p. 7]. For a comparison of different power flow models in the context of expansion planning problems see [NHB22]. In one of the first works on transmission network expansion planning, Garver [Gar70] presents a heuristic for the basic transmission network expansion planning problem.

This basic formulation may be extended in many ways; see the surveys [LR16b,

MSAR18, GYG19] for an overview. Designing transmission networks involves predictions of future requirements and operational constraints, e.g., the future power demand and the power generation by renewable sources. These uncertainties may be modeled via different scenarios, e.g., in [ZCZ17, ZLC19, Gha+21, Muñ+21]. A different way to tackle these problems is robust optimization, where the possible values of stochastic variables is given as so-called uncertainty sets. Robust solutions shall be feasible under all possible realizations of these variables [BBC11]. Robust optimization has been applied successfully to transmission network expansion planning [DA16, Lia+21].

Building storage may be considered together with the grid expansion [BHS18, Hem21]. Moreover, one may include generation expansion planning, i.e., the option to increase the generation capacity of existing generators or to build new generators. This may be modeled as a mixed-integer linear program (MILP) [Li+22].

Expanding the grid may also be combined with allowing lines to be switched off [KSK10, MPS10, MNMR20]. Using the linearized AC power flow this problem can be modeled as an MILP. This setting may further be extended by considering uncertainty [VP12, VBP13].

One may also consider expansion planning with FACTS (flexible AC transmission system). These are devices that allow the operator to influence the power flow on the lines. One study compares building FACTS to adding new transmission lines [BOGR11]. The authors find that building FACTS allows postponing the addition of new transmission lines. Hence, one does not need to commit to building transmission lines early, which allows considering future information when deciding whether and which additional lines are needed. Transmission network expansion planning with the ability to place FACTS can be formulated as an MILP [LPC20].

After the problem formulation has been fixed, it needs to be solved. While the exact solution method depends on the given problem formulation, they typically fall into few classes [GYG19]. Often meta-heuristics such as genetic algorithms are used, in particular if multiple objectives are considered [CM21, p. 33]. They have the advantage that their implementation is typically rather simple and that they compute a set of good solutions instead of only a single one [CM21]. However, they give no guarantees regarding the solution quality. Recent examples of this approach include [ASSR21, Ref+21, Gha+21]. Another common solution technique is formulating the problem as a mathematical program, typically as an MILP or mixed-integer nonlinear program (MINLP) [AMC03, DZMR16, EREG16, DEG17]. An overview over acceleration techniques for MILPs used in transmission network expansion planning problems is given by Lumbreras and Ramos [LR16a]. There are also heuristics based on solving linear programs [NB19]. Occasionally, artificial neural networks are used as well [AE02]. Several of the approaches mentioned above may be combined. For example, a meta-heuristic may be used to find feasible solutions, and an MILP to

find lower bounds [VSC22]. This combines the advantage that meta-heuristics may find good solutions fast with the advantage of MILPs that they provide provable guarantees for the solution qualities.

In most formulations of expansion planning problems, a set of candidate lines is given as an input. However, unlike the topology of the existing grid, the candidate edges are immediately given in the real world, but have to be determined first. Algorithms to automatically determine candidate lines are often based on information from power flow computations in the existing transmission network such as locational marginal prices [LRS14, Wu+18]. To keep the set of candidate lines sufficiently small, the set of candidate lines may need to be pruned [LRS14]. Moreover, new buses may be included as well [Wu+18].

The literature review above shows that grid expansion is an important and well-studied topic. However, the works on this topic have mostly taken a rather applied perspective. Typically, this means making the models more realistic or proposing different algorithms to tackle the problems. While this endeavor is certainly worthwhile, it can be noted that a more theoretical understanding of the problems is still lacking. It is known that the expansion planning problems are $\mathcal{NP}$-hard [MPS10, OR14] in general, but not much more is known. One goal of this thesis is therefore to obtain a deeper theoretical understanding of problems related to grid expansion. The second goal is to identify gaps in the literature on network design problems and to develop algorithms for these problems.

Thesis Outline

We outline the structure of the remainder of this thesis. We would like to point out that parts of this work have previously been published in conference proceedings [WW20, WW21, GW22].

Chapter 2 – Fundamentals: We introduce fundamental concepts and notations we use throughout the thesis. In particular, we define flows and electrical flows, and refer to algorithms to compute them.

Chapter 3 – Complexity of Flow Expansion: We analyze the computational complexity of expanding flow networks—both graph-theoretical and electrical. The results include both $\mathcal{NP}$-hardness proofs and polynomial-time algorithms for special cases.

Chapter 4 – Expanding Electrical Networks to Prevent Critical Edges: We consider mitigations for negative effects caused by failing lines in the power grid. We base our analysis of the severity of line failures on a criterion for critical

edges. We formulate this criterion in the context of network expansion planning problems, and exemplify it by defining two problems with this criterion. Both can be formulated as an MILP, and for one problem, we additionally present a heuristic. Finally, we evaluate the heuristic and the MILP on a set of benchmark instances.

Chapter 5 – Algorithmic Approaches for Microgrid Cable Layouts: We deal with designing the cable layout of a microgrid. We formalize the problem as a geometric optimization problem, similar to the Euclidean Steiner Tree problem. Further, we present a hybrid genetic algorithm for this new problem and evaluate its performance on a set of benchmark instances.

Chapter 6 – FACTS Flows: We study the problem FACTS Flow, which asks whether there is a FACTS flow in a network. A FACTS flow is a generalization of an electrical flow, in which some edge properties may be varied. Unlike the related problem Maximum FACTS Flow of finding a maximum FACTS flow, to the best of our knowledge the complexity of finding any FACTS flow has not been studied before. We transfer the $\mathcal{NP}$-hardness of Maximum FACTS Flow to FACTS Flow and present polynomial-time algorithms for some special cases.

Chapter 7 – Conclusion: We conclude the thesis with a summary, describe how the chapters fit together and point out possible future research directions.

2 Fundamentals

In this chapter we define terms that occur throughout the thesis. We assume that the reader is familiar with basic graph-theoretical terms. An introduction to graph theory can be found in [Die17]. In this thesis we denote the undirected edge between two vertices v and w by $\{v, w\}$ and the directed edge from v to w by vw. Moreover, all cycles in this thesis are simple, i.e., they are connected graphs where all vertices have a degree of 2. As usual, the complete graph on n vertices is denoted by K_n, and the complete bipartite graph with m vertices in one partition and n vertices in the other partition is denoted by $K_{m,n}$.

We further formalize the connection between directed and undirected graphs as follows. Let $G = (V, E)$ be a directed graph. Then, its *underlying undirected graph* $\overline{G}$ is $(V, \overline{E})$, where $\overline{E} = \{\{v, w\} \mid vw \in E\}$. Conversely, an *orientation* of an undirected graph $\overline{G} = (V, \overline{E})$ is a directed graph $G = (V, E)$ such that $\overline{G}$ is the underlying undirected graph of G and for every $\{v, w\} \in \overline{E}$ it either holds $vw \in E$ or $wv \in E$, but not both. For an edge vw we call wv its *reverse* and denote it by $\mathrm{rev}(vw)$. Similarly, for a set E of directed edges, we define $\mathrm{rev}(E) = \{\mathrm{rev}(e) \mid e \in E\}$. The *neighborhood* of a vertex $v \in V$ in a graph G is denoted by $N_G(v)$ and defined by $N_G(v) = \{w \in V \mid vw \in E \text{ or } wv \in E\}$.

2.1 Flows

Flows are fundamental objects that occur in all problems in this thesis. We consider three flow models: graph-theoretical flows, electrical flows, and FACTS flows.

2.1.1 Graph-Theoretical Flows

Undirected flow networks. Conceptually, the graphs on which we compute flows are usually undirected since, e.g., a power line allows for power to be transmitted in either direction. However, flows themselves are inherently directed—there is a difference between a unit flowing from v to w or from w to v along an undirected edge $\{v, w\}$. To capture this direction we always fix an arbitrary orientation G of the undirected graph $\overline{G}$. An *undirected flow network* (or simply a *flow network*) $\mathcal{N}$ is then the directed graph $G = (V, E)$ together with a *demand function* $D\colon V \to \text{Intervals}(\mathbb{R})$ and a capacity function $\text{cap}\colon E \to \mathbb{R}_{>0}$. A *flow* in $\mathcal{N}$ is a function $f\colon E \to \mathbb{R}$ such that

$$|f(vw)| \leq \text{cap}(vw) \qquad \forall vw \in E, \tag{2.1}$$

$$\sum_{u\colon uv\in E} f(uv) - \sum_{w\colon vw\in E} f(vw) \in D(v) \qquad \forall v \in V. \tag{2.2}$$

Equation (2.1) states that the flow on any edge does not exceed the capacity of the edge (called the *capacity constraint*), and Equation (2.2) ensures that the amount of flow entering a vertex v minus the amount leaving v lies within the demand interval of v. The latter property is often called the *conservation of flow* or *flow conservation*. The name can be explained by considering a vertex v with $D(v) = \{0\}$. In this case, Equation (2.2) requires the amount of flow entering v to be equal to the amount of flow leaving v. Given a flow f, we call the term on the left side of Equation (2.2) the *consumption* of v in f, and we denote it by $c_f(v)$. The *total flow value* of a flow f is the sum of all positive consumptions, i.e.,

$$\text{total}(f) = \sum_{v\in V} \max\{c_f(v), 0\} \tag{2.3}$$

Note that values of f may be negative. The sign encodes the direction of the flow: If $f(vw) > 0$, we interpret this as $f(vw)$ units flowing from v to w; if $f(vw) < 0$, we interpret this as $-f(vw)$ units flowing from w to v. Note that the chosen orientation of the underlying undirected graph is arbitrary. Replacing the edge vw with the edge wv and setting the flow value on wv to $-f(vw)$ still yields a flow provided that f is a flow.

A *subnetwork* of a flow network on a graph G is a network on a subgraph G', where the demands and capacities are restricted to the vertices and edges of G'.

Directed flow networks. Occasionally, we need edges that restrict the direction of the flow. Therefore, we define directed flow networks. The definition is similar to the one of undirected flow networks. The two main differences are that we do not require the graph to be an orientation of an undirected graph, and the capacity constraint

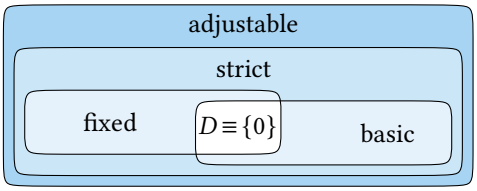


Figure 2.1: The relationship of the demand cases. The only demand function that is both fixed and basic is the function mapping each vertex to $\{0\}$.

bounds the flow values from below by 0. For easier reference we include a complete definition of directed flow networks and flows below, including parts that are the same as in the undirected case. A *directed flow network* is a directed graph $G = (V, E)$ with a demand function $D\colon E \to \text{Intervals}(\mathbb{R})$ and a capacity function $\text{cap}\colon E \to \mathbb{R}_{>0}$. A *flow in a directed flow network* $\mathcal{N}$ is a function $f\colon E \to \mathbb{R}$ such that

$$0 \leq f(vw) \leq \text{cap}(vw) \qquad \forall vw \in E, \tag{2.4}$$

$$\sum_{u\colon uv \in E} f(uv) - \sum_{w\colon vw \in E} f(vw) \in D(v) \qquad \forall v \in V. \tag{2.5}$$

Every undirected flow network can be transformed into an equivalent directed flow network by adding the reverse of every edge with the same capacity. That is, if there is an edge vw with capacity $\text{cap}(vw)$ in the undirected flow network, then the equivalent directed flow network contains both vw and wv, each with capacity $\text{cap}(vw)$.

Special cases. An important special case of a flow network occurs when all demand intervals contain only one element, i.e., there is a function $d\colon V \to \mathbb{R}$ such that $D(v) = \{d(v)\}$ for all $v \in V$. We then say that the flow network has *fixed demands*. This setting is very common in the literature. For example, the standard Minimum Cost Flow problem is phrased in this setting [AMO93]. Note that if the sum of all demands is not 0, there is no flow in the network. When studying properties of networks with fixed demands, we therefore typically require the sum of the demands to be 0. For clarity, we may refer to the general case of demand intervals as networks with *adjustable demands*. Figure 2.1 depicts the relationship between the special cases of the demand choices.

Another special case are flow networks where no demand interval contains both positive and negative values. We call such flow networks *strict*. In particular, all flow networks with fixed demands are strict. In strict networks we can partition the vertex

set V into a set V_S of *sources*, a set V_T of *sinks*, and a set V_I of *intermediate vertices* by defining

$$V_S = \{v \in V \mid \min D(v) < 0\},$$
$$V_T = \{v \in V \mid \max D(v) > 0\},$$
$$V_I = \{v \in V \mid D(v) = \{0\}\}.$$

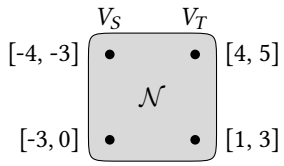
Finally, a strict network is *basic* if every demand interval contains 0.

Computing flows. Computing a flow in a network can be formulated as a linear program. A linear program with constraint matrix A can be solved in $\widetilde{\mathcal{O}}((\mathrm{nnz}(A) + \mathrm{rank}(A)^2)\sqrt{\mathrm{rank}(A)}\log(1/\varepsilon))$ time, where $\mathrm{nnz}(A)$ is the number of non-zero entries of A, $\mathrm{rank}(A)$ is the rank of A, and ε is the desired approximation factor [LS14, LS19]. Formulating the constraints in Equations (2.1) and (2.2) as a linear program with matrix A, we have $\mathrm{nnz}(A) \in \mathcal{O}(m+n)$ and $\mathrm{rank}(A) \in \mathcal{O}(m+n)$, where n is the number of vertices and m the number of edges. Hence, flows can be computed using standard linear program solving techniques in $\widetilde{\mathcal{O}}((m+n)^{5/2}\log(1/\varepsilon))$ time. We shall see in Corollary 2.2 that if the capacities and the boundaries of the demand intervals are integers, there is a flow with integral values. We may set $\varepsilon = \alpha/((m+n)U)$, where U is an upper bound on the input values and α is a sufficiently small constant. This way, we are able to guarantee that rounding the resulting flow values to the nearest integers always yields an (exact) flow. The running time then lies in $\widetilde{\mathcal{O}}((m+n)^{5/2})$; note that the additional factor of $\log(1/\varepsilon) = \log((m+n)U)$ is hidden in $\widetilde{\mathcal{O}}(\cdot)$.

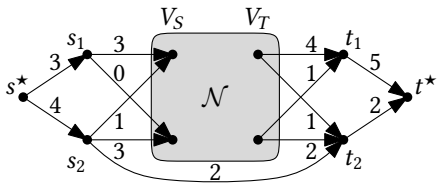
However, we can reduce the flow computation in the setting of adjustable demands to the standard problem of computing a maximum flow from a single source to a single sink in a flow network. For fixed demands such a reduction is well-known and works by adding a super source and a super sink that are connected to all sources and all sinks, respectively; see,e.g., [AMO93, Sec. 6.2]. Below, we present a reduction of the general case of adjustable demands.

Formulated in terms of the notation introduced in this thesis, the Maximum Flow problem is defined as follows. The input consists of a directed flow network $\mathcal{N}$ and two distinguished vertices s and t. The demand of the source s is $(-\infty, 0]$, the demand of the sink t is $[0, \infty)$, and for all other vertices v it holds that $D(v) = \{0\}$. We call networks that fulfill these conditions *s-t-networks*. The goal is then to find a flow in $\mathcal{N}$ that maximizes the consumption of t. Note that in this case the consumption of t in a flow f is then the total flow value of f.

Lemma 2.1. *Let $\mathcal{N}$ be a flow network on a graph with n vertices and m edges. There is a directed s-t-network $\mathcal{N}'$ with $n+6$ vertices and $\Theta(m)$ edges as well as some $a \in \mathbb{R}_{\geq 0}$ such that there is a flow in $\mathcal{N}$ if and only if there is a flow f' in $\mathcal{N}'$ with total flow value a.*



(a) A sketch of the input network showing only the sources and sinks with their demands.



(b) The corresponding directed s-t-network with source $s^\star$ and sink $t^\star$. The values at the edges represent the capacities.

Figure 2.2: An example of the construction in the proof of Lemma 2.1.

Proof. Let $\mathcal{N}$ be a flow network on the graph $G = (V, E)$. We may assume without loss of generality that $\mathcal{N}$ is strict by splitting the demand intervals that contain both positive and negative values and assigning the two intervals to two new vertices. More precisely, let v be a vertex such that $D(v)$ contains both positive and negative values. Then, we add two new vertices s_v and t_v with edges $s_v v$ and $v t_v$ of sufficient capacity. We set $D'(v) = \{0\}$, $D'(s_v) = [\min D(v), 0]$, and $D'(t_v) = [0, \max D(v)]$. Repeating this process, we obtain a network, where every demand interval either contains only non-positive values (at sources) or non-negative values (at sinks). Let V_S be the set of sources and V_T be the set of sinks.

The key idea of the construction below is to route flow from a super source to an original source s of $\mathcal{N}$ via two different paths. The first path (via a vertex s_1) conducts the minimum amount of flow that is necessary to fulfill the demand constraint of s, i.e., $-\max D(s)$. The second path (via a vertex s_2) optionally conducts flow until the lower bound of the consumption of s is reached, i.e., up to $\max D(s) - \min D(s)$. By choosing the capacities appropriately, we ensure that in any maximum flow the edges of the first path are saturated. A similar construction is added for all other sources and all sinks.

More precisely, we define the graph $G' = (V', E')$ of the Maximum Flow instance by

$$
\begin{aligned}
V' &= V \cup \{s^\star, s_1, s_2, t^\star, t_1, t_2\} \\
E_S &= \{s^\star s_1, s^\star s_2\} \cup \{s_1 s, s_2 s \mid s \in V_S\}, \\
E_T &= \{t_1 t^\star, t_2 t^\star\} \cup \{t t_1, t t_2 \mid t \in V_T\}, \\
E' &= E \cup \operatorname{rev}(E) \cup E_S \cup E_T \cup \{s_2 t_2\}.
\end{aligned}
$$

An example of the graph constructed in this way can be found in Figure 2.2.

In order to define the capacities of the edges, we use bounds for the sum of the consumptions in any flow in $\mathcal{N}$. Let

$$\hat{d}_{\max} = \min\left\{\sum_{s\in V_S} -\min D(s), \sum_{t\in V_T} \max D(t)\right\},$$
$$\hat{d}^S_{\min} = \sum_{s\in V_S} -\max D(s),$$
$$\hat{d}^T_{\min} = \sum_{t\in V_T} \min D(t).$$

Observe that $\hat{d}_{\max}$ is an upper bound on the sum of the consumptions of all sinks in any flow in $\mathcal{N}$, and $\hat{d}^S_{\min}$ and $\hat{d}^T_{\min}$ are lower bounds. Moreover, $\hat{d}^T_{\min}$ is the sum of the minimum consumptions of all sinks. Similarly, $\hat{d}^S_{\min}$ is the sum of the minimum productions (i.e., the negative of the consumption) of the sources. We define the capacities of the edges by

$$\operatorname{cap}'(vw) = \begin{cases} \operatorname{cap}(vw), & vw \in E, \\ \operatorname{cap}(wv), & vw \in \operatorname{rev}(E), \\ -\max D(s), & vw = s_1 s \text{ for some } s \in V_S, \\ \max D(s) - \min D(s), & vw = s_2 s \text{ for some } s \in V_S, \\ \hat{d}^S_{\min}, & vw = s^\star s_1, \\ \hat{d}_{\max} - \hat{d}^S_{\min}. & vw = s^\star s_2, \\ \min D(t), & vw = t t_1 \text{ for some } t \in V_T, \\ \max D(t) - \min D(t), & vw = t t_2 \text{ for some } t \in V_T, \\ \hat{d}^T_{\min}, & vw = t_1 t^\star, \\ \hat{d}_{\max} - \hat{d}^T_{\min}, & vw = t_2 t^\star, \\ \min\{\hat{d}_{\max} - \hat{d}^S_{\min}, \hat{d}_{\max} - \hat{d}^T_{\min}\}, & vw = s_2 t_2. \end{cases}$$

Let the resulting directed flow network be $\mathcal{N}'$. We claim that there is a flow with total flow value $\hat{d}_{\max}$ in $\mathcal{N}'$ if and only if there is a flow in $\mathcal{N}$.

Let f be a flow in $\mathcal{N}$. It can be extended to a flow f' in $\mathcal{N}'$ as follows. Send the maximum amount of flow along all edges incident to $s^\star$, s_1, $t^\star$, and t_1. Then, send the required amount of flow along edges from s_2 to sources in $\mathcal{N}$ to match the consumption of the sources, i.e., set $f'(s_2 s) = \max D(s) - c_f(s)$ for $s \in V_S$. Similarly, set $f'(t t_2) = c_f(t) - \min D(t)$ for all sinks $t \in V_T$. Finally, route $\hat{d}_{\max} - \sum_{t\in V_T} c_f(t)$ units of flow along $s_2 t_2$. It can be verified that f' defined in this way is a flow in $\mathcal{N}'$ with a total flow value of $\hat{d}_{\max}$.

Conversely, let f' be a flow in $\mathcal{N}'$ with total flow value $\hat{d}_{\max}$. Let $f\colon E \to \mathbb{R}$ be defined by $f(vw) = f'(vw) - f'(wv)$ for all $vw \in E$. Essentially, f describes the restriction of f' to the edges in the undirected network $\mathcal{N}$. We claim that f is a flow in $\mathcal{N}$. To prove this claim, we need to verify that the consumptions of the vertices lie within the demand intervals. For an intermediate vertices v we have $c_f(v) = c_{f'}(v) = 0$. For the sinks, we observe that the cut formed by the edge $t_2t^\star$ and all edges tt_1 for $t \in V_{\mathrm{T}}$ has a size of $\hat{d}_{\max}$. By the max-flow min-cut theorem (see e.g., [Die17, Thm. 6.2.2]) the edges across the cut must be saturated by f'. Hence, the consumption of a sink t in f is at least $\mathrm{cap}'(tt_1) = \min D(t)$. By construction, we have $\mathrm{cap}'(tt_1) + \mathrm{cap}'(tt_2) = \max D(t)$. Hence, we have $c_f(t) \leq \max D(t)$ as well, showing $c_f(t) \in D(t)$. A similar argument considering the cut formed by the edge $s^\star s_2$ and all edges $s_1 s$ for $s \in V_{\mathrm{S}}$ yields $c_f(s) \in D(s)$ for all $s \in V_{\mathrm{S}}$. □

This lemma shows that the computation of flows in the setting defined in this thesis can be reduced to computing maximum flows in directed s-t-networks. This is a classical problem, which has received a lot of attention in the recent years. A maximum flow in a directed s-t-network with n vertices, m edges, integral capacities and maximum capacity U can be computed in $\widetilde{\mathcal{O}}((m + n^{3/2}) \log U)$ time [Bra+21b], in $\widetilde{\mathcal{O}}(m^{3/2-1/58} \log U)$ time [Bra+21a, Bra+22], and in $m^{4/3+o(1)}U^{1/3}$ time [KLS22, Kat20, LS20a]. There is also a randomized algorithm that computes such a flow in $m^{11/8+o(1)}U^{1/4}$ time with high probability [LS20b]. Interestingly, there is a series of works, starting with an $\widetilde{\mathcal{O}}(m^{10/7}U^{1/3})$-time algorithm [Mąd16] and including the $m^{11/8+o(1)}U^{1/4}$-time algorithm [LS20b], that use the computation of electrical flows as building blocks in their algorithms.

For directed s-t-flow networks with integral capacities the integral flow theorem guarantees that there is a maximum flow where all values are integers [Die17, Cor. 6.2.3]. Such flows are called *integral*. By Lemma 2.1 the same holds for flows in the setting we describe.

Corollary 2.2. *If there is a flow in a flow network $\mathcal{N}$ with integral capacities and integral demand interval endpoints, then there is an integral flow in $\mathcal{N}$.*

2.1.2 Electrical Flows

To model the flow of electric power we use a *linearized AC power flow model*, which is sometimes called the *DC power flow model* [FR16]. As the name suggests, it is derived from the AC power flow model by applying suitable simplifications [FR16]. In this model a unit of flow represents a unit of power (measured in W) that is transmitted. Every edge has an additional parameter, called the *susceptance*, which is represented by a function $b\colon E \to \mathbb{R}_{>0}$. We call a network with this additional edge parameter

an *electrical network*. The model enforces an additional constraint called *Kirchhoff's voltage law*. A flow f satisfies Kirchhoff's voltage law if there is a function $\theta\colon V \to \mathbb{R}$ such that

$$f(vw) = b(vw) \cdot (\theta(v) - \theta(w)) \qquad \forall vw \in E. \tag{2.6}$$

Based on the derivation from AC power flow, the values of θ represent *(power) angles*. We call any flow that satisfies Equation (2.6) an *electrical flow*.

Equivalently (and how they were originally stated [Kir47]), the constraints in Equation (2.6) can be expressed in terms of restrictions of flows in cycles [Bol98, Koc+16]. Let C be a cycle of the underlying undirected graph $\overline{G}$ of G. We fix an orientation of the cycle by first choosing the direction of one edge arbitrarily, and then walking along the cycle in the chosen direction. We note that the direction of an edge in the directed graph G and in the orientation of C may differ. Let $\delta^C_{vw} = +1$ if vw is oriented the same in both cases, and $\delta^C_{vw} = -1$ otherwise. A flow is electrical if and only if for every oriented cycle C in $\overline{G}$ it holds that

$$\sum_{vw \in E(C)} \frac{f(vw) \cdot \delta^C_{vw}}{b(vw)} = 0.$$

Note that Kirchhoff [Kir47, Kir58] formulated this law in the context of resistor networks, which can be described as graphs where each edge vw has a resistance $r(vw)$. DC flows in these resistor networks need to satisfy $f(vw) = r(vw) \cdot (\pi(v) - \pi(w))$ for some *potential* function π. Physically, the variables that occur in this equation and the ones in Equation (2.6) have different meanings. Mathematically, however, the two equations are equivalent by setting $b(vw) = 1/r(vw)$ for all edges vw and $\theta(v) = \pi(v)$ for all vertices v. Most literature on electrical flows, e.g., [Bol98, Chr+11], uses the resistance network model. But their results can be directly transferred to the model we use by applying the observation just mentioned.

In networks with fixed demands the flow conservation (Equation (2.2)) and Kirchhoff's voltage law (Equation (2.6)) already uniquely define the flow values [Bol98, p. 29]. That means that we can use these equations to compute the flow values for all edges. We can then determine whether this is actually an electrical flow by checking whether no edge capacities are exceeded. The uniqueness of the solution implies that there is at most one electrical flow in an electrical flow network. Moreover, if the demand sum is 0 and the capacities are sufficiently large, there is always exactly one electrical flow.

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be two networks on the same graph G with infinite capacities and fixed demands given by d_1 and d_2, respectively. We assume that the demand sums in both networks is 0; otherwise, there is no flow. Let f_i be the uniquely determined

electrical flow in $\mathcal{N}_i$. Then, $f_1 + f_2$ defines an electrical flow in a network on G with demands $d_1 + d_2$. In other words, the addition of two electrical flows is an electrical flow as well.

Conversely, an electrical flow f in a network with n vertices can be decomposed into electrical flows $f_1, \ldots, f_{n-1}$ such that $f = f_1 + \cdots + f_{n-1}$ and f_i is a flow in a network $\mathcal{N}_i$ with exactly one source and exactly one sink. All other vertices have a demand of 0. Moreover, the networks $\mathcal{N}_i$ can be computed from $\mathcal{N}$ in $\mathcal{O}(n^2)$ time. This decomposition allows us to reduce the computation of electrical flows in networks with fixed demands to $n - 1$ electrical flow computations in networks with one source and one sink. For example, in a cycle one such computation is possible in linear time. Thus, an electrical flow in a cycle with fixed demands can be computed in quadratic time.

Computing electrical flows. The constraints for electrical flows can be formulated as a linear program. Using the same general linear programming solving techniques we mention for graph-theoretical flows [LS14, LS19], electrical flows can be computed in $\widetilde{\mathcal{O}}((m+n)^{5/2})$ time, where n is the number of vertices and m the number of edges in the network.

If the demands are fixed, the check whether the consumptions lie within the demand intervals (Equation (2.2)) become equality checks. Moreover, the variables representing the flow values can be eliminated using Equation (2.6). In total, we obtain the following system of linear equations.

$$\sum_{u\,:\,uv\in E} b(uv)(\theta(u) - \theta(v)) - \sum_{w\,:\,vw\in E} b(vw)(\theta(v) - \theta(w)) = d(v) \quad \forall v \in V. \tag{2.7}$$

In fact, this system can be written in terms of the *Laplacian* L of the graph (with edge weights given by b) as $L\theta = d$, where θ and d are interpreted as vectors [Bol98]. The Laplacian L of a graph is symmetric diagonally dominant (SDD), i.e., we have $L = L^{\mathrm{T}}$, and $L_{ii} \geq \sum_{j\neq i} |L_{ij}|$ for all $i \in \{1, \ldots, n\}$ [KOSZ13]. Such systems can be solved approximately in nearly linear time. Here, an *ε-approximate solution* $\tilde{\theta}$ satisfies $\|L\tilde{\theta} - d\| < \varepsilon$ and $\|\tilde{\theta} - \theta\|_L \leq \varepsilon \|\tilde{\theta}\|_L$, where θ is the exact solution and $\|x\|_L = \sqrt{x^{\mathrm{T}} L x}$. The first such algorithm was presented by Spielman and Teng [ST04]. Following this breakthrough, there is a long line of works improving upon this result; see [ST14] for a history of the developments. Recently, it was shown that ε-approximate solution of SDD systems can be computed with high probability in expected $\mathcal{O}(m\sqrt{\log n}(\log\log n)^{3+\delta}\log(1/\varepsilon))$ time for every $\delta > 0$ [Coh+14].

Note that in Equation (2.7) the capacity constraints of Equation (2.1) are neglected. However, any two solutions θ_1 and θ_2 of Equation (2.7) have a constant difference, i.e., there is some constant k such that $\theta_1(v) = \theta_2(v) + k$ for all $v \in V$ [Bol98]. As the

flow on an edge depends only on the difference of the angles at the incident vertices, all solutions induce the same flow. Hence, it suffices to check afterwards whether this flow satisfies all capacities.

Besides these numerical algorithms there are also combinatorial algorithms for computing approximate electrical flows in networks with fixed demands. These methods have the advantage that they do not require the heavy machinery used for solving and are thus much simpler to implement. The first such algorithm by Kelner et al. [KOSZ13] uses a low-stretch spanning tree, i.e., for $vw \in E$ the ratio of the length (given by the inverse b^{-1} of the susceptance) of the unique path from v to w along the tree to the length of vw is small. Then, an initial flow is computed, which is iteratively updated via simple modifications. Every such flow f_i in iteration i induces angles θ_i. During the algorithm the property that f_i is indeed a flow is maintained, i.e., f_i satisfies the flow conservation constraint (Equation (2.2)), but f_i and θ_i do not necessarily satisfy Kirchhoff's voltage law (Equation (2.6)). The goal of each update step is to reduce the violation of this constraint. In total, the algorithm computes an ε-approximate electrical flow in $\mathcal{O}(m \log^2 n \log\log n \log(n/\varepsilon))$ time. This algorithm has also been implemented and evaluated experimentally [HLMW16]. It was found that the results confirm the predicted asymptotic behavior, but the constant factors are large such that it is not competitive with other methods in practice.

Henzinger, Jin, and Williamson [HJW20] present an algorithm that is dual to the one by Kelner et al. [KOSZ13]. It also updates functions f_i and θ_i, but it maintains the property that Kirchhoff's voltage law is satisfied, whereas the flow conservation may be violated. They achieve a running time of $\mathcal{O}(mn \log n \log\log n \log(1/\varepsilon))$. This algorithm can be adapted to p-norm flows, which are a generalization of electrical flows [HJPW21].

2.1.3 FACTS Flows

Electrical flows can be generalized by loosening the restriction on the susceptances of the edges. A straight-forward way to do this is to allow the susceptances to be adjusted within a certain range. The model is inspired by and named after flexible AC transmission system devices, which allow the transmission grid operator to influence the impedance of power lines [Hin93, HH14].

The model we use in this thesis is based on the work by Lehmann, Bent, and Pan [LBP15]. To be self-contained and consistent with the notation in the previous sections, we restate the definition here. Intuitively, the key difference to electrical flow networks and electrical flows is that the susceptance of the edges is no longer fixed, but instead each edge $e \in E$ has an interval $B(e)$, from which the susceptance may be chosen. Formally, a *FACTS flow network* $\mathcal{N}$ consists of an orientation $G = (V, E)$ of an undirected graph, a *demand function* $D\colon V \to \text{Intervals}(\mathbb{R})$, a *capacity*

function cap: $E \to \mathbb{R}_{\geq 0}$, and a *susceptance interval function* $B\colon E \to \text{Intervals}(\mathbb{R}_{>0})$.

A *FACTS flow* in the FACTS flow network $\mathcal{N}$ is a flow f for which there are functions $b\colon E \to \mathbb{R}_{>0}$ and $\theta\colon V \to \mathbb{R}$ such that for all edges $vw \in E$ it holds that

$$b(vw) \in B(vw), \tag{2.8}$$
$$f(vw) = b(vw) \cdot (\theta(v) - \theta(w)). \tag{2.9}$$

In other words, we can assign susceptances from the susceptance intervals to the edges such that f is an electrical flow in the resulting electrical flow network.

It is known that maximizing the flow value of a FACTS flow is $\mathcal{NP}$-hard [LBP15]. More precisely, it is strongly $\mathcal{NP}$-complete to determine whether a given FACTS flow network admits a FACTS flow of value at least k, where $k \in \mathbb{R}_{\geq 0}$ is part of the input. We show in Chapter 6 how to transfer this hardness to computing any FACTS flow in a network.

2.2 Expansion of Flow Networks

In expansion problems of flow networks, the input contains a flow network $\mathcal{N}$ on a graph $G = (V, E)$, where the edge set is partitioned into a set E_{ex} of *existing edges* and a set E_{cand} of *candidate edges* (or *candidates* for short). In addition to the capacities and possibly susceptances, there are costs associated with the candidates. We represent them by a function $c\colon E_{\text{cand}} \to \mathbb{R}_{\geq 0}$. The *existing network* is the subnetwork on the graph (V, E_{ex}), i.e., it contains only the existing edges.

An *expansion* $\mathcal{H}$ of the network $\mathcal{N}$ is a subnetwork of $\mathcal{N}$ on a graph $H = (V, E_H)$ such that $E_{\text{ex}} \subseteq E_H \subseteq E$. The *total cost* of an expansion $\mathcal{H}$ is defined by

$$\text{cost}(\mathcal{H}) = \sum_{e \in E_H \setminus E_{\text{ex}}} c(e).$$

An expansion $\mathcal{H}$ of an (electrical) network $\mathcal{N}$ is *feasible* if there is an (electrical) flow in $\mathcal{H}$.

We note that defining an expansion as a subnetwork seems to be counter-intuitive. However, the network contains both the existing and all candidate edges. As not all candidate edges need to be chosen, an expansion $\mathcal{H}$ is a subnetwork of $\mathcal{N}$. But by definition, every expansion contains the existing network as a subnetwork. In this sense, $\mathcal{H}$ expands the existing network.

We call the problem of finding a feasible expansion of a (non-electrical) network with minimum total cost Flow Expansion. The analogous problem of finding a feasible expansion of an electrical network with minimum total cost is Electrical

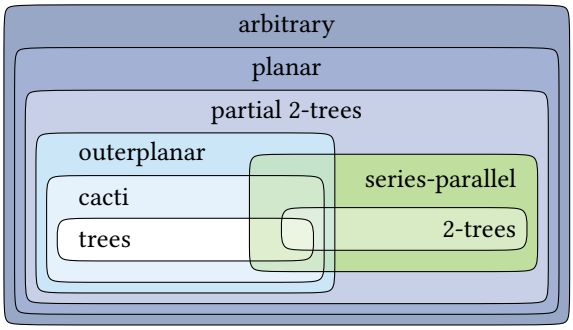


Figure 2.3: The relationships of the graph classes. A box for a graph class $\mathcal{C}_1$ is contained in another for $\mathcal{C}_2$ if and only if $\mathcal{C}_1$ is contained in $\mathcal{C}_2$.

Flow Expansion. A feasible expansion of minimum total cost is called an *optimal expansion*.

For both problems we also define the corresponding *decision variant* as: Given some $k \in \mathbb{R}_{\geq 0}$, is there a feasible expansion with total cost at most k?

2.3 Graph Classes

In this thesis we classify the complexity of problems on flow networks based on the graph class the underlying graphs belong to. In this section, we give an overview of the graph classes that are relevant to our results. The relationships between the classes are illustrated in Figure 2.3. Note that there are many more graph classes than we cover here. We refer the interested reader to [BLS99] for a huge list of graph classes and their relationships. The following list of graph classes are ordered from largest to smallest.

Arbitrary graphs. The largest class we consider is of course the class of all graphs. In our results, we denote them by *arbitrary graphs*.

Planar graphs. The next smaller class contains the *planar graphs*. These are the graphs that can be embedded into the plane without any edge crossings. A famous result is that a graph is planar if and only if it is K_5- and $K_{3,3}$-minor-free [Wag37]. Planar graphs can be recognized in linear time [CNAO85].

Partial 2-trees. We further consider the class of *partial 2-trees*, which are the subgraphs of 2-trees. A *2-tree* is recursively defined as follows. The graph K_2 is a 2-tree. Further, adding a vertex with edges to the endpoints of one edge of a 2-tree yields another 2-tree [BLS99]. Partial 2-trees are precisely the K_4-minor-free graphs [WC83]. Equivalently, these are the subgraphs of *series-parallel graphs* [Duf65]. We frequently use the latter characterization as the structure of a series-parallel graph can be described by a tree structure, which is very helpful when formulating algorithms for series-parallel graphs.

Formally, a *two-terminal series-parallel graph* is a graph with two vertices designated as *terminals*: one *source* and one *sink* such that it can be obtained via the following operations. First, a single edge is a two-terminal series-parallel graph. Then, let G_1 and G_2 be two-terminal series-parallel graphs with sources s_1 and s_2 and sinks t_1 and t_2. The *series composition* of G_1 and G_2 is obtained by merging t_1 with s_2, forming a new graph with source s_1 and sink t_2. The *parallel composition* of G_1 and G_2 is obtained by merging the two sources (forming the new source) and the two sinks (forming the new sink). A *series-parallel graph* then is a graph G for which two vertices s and t can be chosen such that G is a two-terminal series-parallel graph with terminals s and t.

We can represent the way the compositions are applied to get a graph G by a rooted binary tree: the *sp-tree* $\mathcal{T}$ [BLS99]. The leaves of the tree correspond bijectively to the edges of G. The inner nodes correspond to the two operations above. The *s-node* corresponds to the series composition of its two children, and the *p-node* corresponds to the parallel composition. It is possible to compute the sp-tree of a series-parallel graph both with and without fixed terminals in linear time [BA01]. The case without fixed terminals can actually be reduced to the case with fixed terminals in linear time [He91, Epp92].

Any partial 2-tree can be completed to a 2-tree in linear time [WC83]. Given a partial 2-tree G, we can thus compute an sp-tree of a supergraph of G in linear time: First, complete G to a 2-tree $G^\star$. Since all 2-trees are series-parallel, we can then compute an sp-tree for $G^\star$.

Outerplanar graphs. A graph is *outerplanar* if it can be embedded into the plane such that all vertices lie on the outer face. The outerplanar graphs can be characterized by being K_4- and $K_{2,3}$-minor-free [CH67]. Hence, they form a strict subclass of the partial 2-trees. Outerplanar graphs can be recognized in linear time [Mit79].

Cacti. A special case of outerplanar graphs are *cacti*. A cactus is a graph in which any two simple cycles share at most one vertex. Hence, every biconnected component of a cactus is either a single edge or a simple cycle. Thus, cacti can be recognized in

linear time by computing the biconnected components [HT73] and then checking whether they are edges or simple cycles.

Trees. Finally, a graph is a *tree* if it is connected and does not contain any cycles. This can be checked in linear time with a depth-first search. A special case of trees are *stars*, which are graphs of the form $K_{1,n}$ for $n \in \mathbb{N}$. That is, they have one central vertex which is adjacent to all other vertices.

2.4 Structures in Graphs

Let G be a connected graph. If for a vertex v the graph $G - v$ after removing v is not connected, then v is called a *cut-vertex*. Similarly, an edge e whose removal disconnects the graph is called a *bridge*. A graph is *biconnected* if it has no cut-vertices, i.e., the graph is still connected after any single vertex is removed. A maximal biconnected subgraph of a graph is called a *biconnected component* or *block*. We observe that by this definition every edge belongs to exactly one biconnected component. However, a vertex v may belong to multiple biconnected components. In this case, v is a cut-vertex. In fact, the converse is true as well. Every cut-vertex belongs to at least two biconnected components.

The structure of the biconnected components and cut-vertices of a graph can be described by a *block-cut tree*. It has one node per cut-vertex and one node per biconnected component. There is an edge between a node for a cut-vertex v and a node for a biconnected component H if and only if v belongs to H. The block-cut tree of a graph can be computed in linear time [HT73].

3 Complexity of Flow Expansion

In this chapter we analyze the computational complexity of Flow Expansion and Electrical Flow Expansion under various restrictions. In particular, we consider the structures of the underlying graphs of the networks and the demands types. For adjustable demands both problems are already $\mathcal{NP}$-hard if the network is a tree. For the case of fixed demands, we give polynomial-time algorithms for networks on cacti, and we prove that the problems become $\mathcal{NP}$-hard on planar 2-trees. An important difference between Flow Expansion and Electrical Flow Expansion is that for Flow Expansion it can be decided in polynomial time whether there is an expansion, whereas for Electrical Flow Expansion this is $\mathcal{NP}$-hard to decide in general.

This chapter is based on joint work with Dorothea Wagner [WW21].

3.1 Introduction

Expanding flow networks is a challenging task with a wide range of applications such as deciding where to build new roads, which regions to connect by new rail lines, or where to build new power lines for transmitting electrical power. The latter problem is often called Transmission Network Expansion Planning (TNEP or TEP), and a huge body of research on it exists in the electrical engineering community; see for example [MSAR18, GYG19] for recent surveys and [LAR21] for a book that introduces the topic. What distinguishes Transmission Network Expansion Planning from other flow expansion problems is the underlying flow model, which needs to capture additional physical laws such as Kirchhoff's voltage law.

Many variants of Transmission Network Expansion Planning are considered in the literature. Among other differences, they differ in the underlying power flow model (in particular, AC vs. linearized AC power flow), the focus on reliability, regulated vs. deregulated energy markets, how they deal with the uncertainty of future events, or whether they consider one time period or multiple. The focus in most works lies on having more realistic models of the real-world transmission system, e.g., uncertainties in the properties of the equipment [ZLC19] or finding more efficient solution methods [NB19, MKMC22, Zop+22] especially for Transmission Network Expansion Planning based on an AC power flow model [MC21, VSC22].

Due to this large range the term Transmission Network Expansion Planning is somewhat ambiguous, and we choose to frame our results in terms of the Electrical Flow Expansion problem defined in Section 2.2. More precisely, Electrical Flow Expansion is a formalization of a very basic version of Transmission Network Expansion Planning. There is only one time period, the demand intervals are assumed to be known, and we use the linearized AC power flow model. In accordance with our definitions in Section 2.1 we call flows adhering to the linearized AC power flow model *electrical flows*.

For these problems few theoretical results are established in the literature. It is known that Electrical Flow Expansion is $\mathcal{NP}$-hard [MPS10]. The related problem Maximum Transmission Switching Flow, where edges may be removed instead of added, is $\mathcal{NP}$-hard as well even if the underlying graph is series-parallel [Koc+16, Gra+18]. On general graphs it is $\mathcal{NP}$-hard to decide whether there is a subnetwork that admits an electrical flow [LGV14]. These results can easily be transferred to Electrical Flow Expansion.

In this chapter we thoroughly analyze the complexity of Electrical Flow Expansion and its cousin Flow Expansion, where graph-theoretical flows are considered instead of electrical flows. Recall that an instance of these problems consists of an (electrical) flow network where the edge set of the underlying graph is partitioned into *existing* and *candidate* edges. There are costs associated with the candidate edges. The goal is to find the cheapest subset of candidate edges to add to the existing network such that the resulting expansion admits an (electrical) flow satisfying all constraints. We further assume in this chapter that all input values are integral. However, note that we do not make this assumption for the flows in the resulting expansion.

Besides Electrical Flow Expansion, which is based on an electrical flow, we consider the expansion of graph-theoretical flow networks as well. We call the expansion problem for graph-theoretical flow networks Flow Expansion. The problem Flow Expansion can be considered as a minimum-cost flow problem with non-linear cost functions. For the related problem of maximizing the flow subject to a budget constraint heuristics exist [EF82]. A generalization of Flow Expansion

Table 3.1: Overview over the complexity of the expansion planning problems. [a] The hardness already applies to the feasibility variant; see Proposition 3.4 for details. [b] This result is not stated explicitly; see Proposition 3.9 for more details.

	Flow Expansion		Electrical Flow Expansion	
Graph class	fixed	adjustable	fixed	adjustable
Tree	$\mathcal{O}(n)$ time (Prop. 3.11)	$\mathcal{NP}$-compl. (Thm. 3.5)	$\mathcal{O}(n)$ time (Prop. 3.11)	$\mathcal{NP}$-compl. (Cor. 3.6)
Cactus	$\mathcal{O}(n^2)$ time (Thm. 3.14)	$\mathcal{NP}$-compl. (Thm. 3.5)	$\mathcal{O}(n^2)$ time (Thm. 3.14)	$\mathcal{NP}$-compl.[a] [LGV14]
Partial 2-tree	$\mathcal{NP}$-compl. (Cor. 3.8)	$\mathcal{NP}$-compl. (Thm. 3.5)	$\mathcal{NP}$-compl.[a] [Koc+16]	$\mathcal{NP}$-compl.[a] [LGV14]
Planar	strongly $\mathcal{NP}$-compl. [MPS10]	strongly $\mathcal{NP}$-compl. [MPS10]	strongly $\mathcal{NP}$-compl.[a] [LGV14]	strongly $\mathcal{NP}$-compl.[a] [LGV14]
Arbitrary	$\mathcal{APX}$-hard [MPS10][b]	$\mathcal{APX}$-hard [MPS10][b]	strongly $\mathcal{NP}$-compl.[a] [LGV14]	strongly $\mathcal{NP}$-compl.[a] [LGV14]

is the Fixed Charge Transportation Problem, where the cost of each edge with non-zero flow is given by a fixed amount plus an amount proportional to the flow on the edge. It is $\mathcal{NP}$-hard as well, and there are several exact algorithms based on integer linear programming [RBM15, MR18].

Contribution. We give a fine-grained view on the complexity of various variants of (Electrical) Flow Expansion. To this end we classify the variants of (Electrical) Flow Expansion according to the graph classes, the demand type (adjustable vs. fixed demands), the number of sources and sinks, and the flow type.

Table 3.1 gives an overview over the computational complexities of variants of the flow network expansion problems. Note that we only include two of the four demand types defined in Section 2.1. We argue in Section 3.2 why the other two types can be ignored. It includes both existing results and results that were first described in the paper this chapter is based on [WW21]. Some cases for which we report existing results actually consider different but very similar problems, or they are not stated

Table 3.2: Overview over the complexity of the expansion planning problems in two-terminal networks. The class sps-graphs refers to a subclass of two-terminal series-parallel graphs, which is defined in Section 3.6. [a] The hardness already applies to the feasibility variant.

Graph class	Flow Expansion		Electrical Flow Expansion	
Tree	$\mathcal{O}(n)$ time (Prop. 3.11)		$\mathcal{O}(n)$ time (Prop. 3.11)	
Cactus	$\mathcal{O}(n^2)$ time (Thm. 3.14)		$\mathcal{O}(n^2)$ time (Thm. 3.14)	
Sps-graph	$\mathcal{NP}$-compl. (Cor. 3.8)	pseudo-poly. (Prop. 3.17)	$\mathcal{NP}$-compl. (Thm. 3.7)	pseudo-poly. (Prop. 3.17)
Two-terminal series-parallel	$\mathcal{NP}$-compl. (Cor. 3.8)	$\mathcal{O}(n^3C^2)$ time (Prop. 3.16)	$\mathcal{NP}$-compl.[a] [Gra+18]	

explicitly, but they are easy consequences of known results. In some cases we feel that stating the arguments explicitly is beneficial. These cases are marked in the table, and references to the corresponding propositions are given.

It can be noted that the existing results only include hardness proofs. This is not suprising as the table indicates that the expansion problems are already $\mathcal{NP}$-hard on moderately complex graph classes such as partial 2-trees, and realistic networks typically have more complex structures [KHCM18]. Our results extend the $\mathcal{NP}$-hardness to networks on trees if the demands are adjustable (Theorem 3.5).

For more restricted variants, e.g., cacti with fixed production (Theorem 3.14), we give polynomial-time algorithms. To the best of our knowledge these are the first polynomial-time algorithms for any variant of (Electrical) Flow Expansion.

In addition, we study the case that the networks have only one source and one sink; see Table 3.2. We prove that in this setting networks with fixed demands and networks with adjustable demands are equivalent. Therefore, we do not distiguish between these two in Table 3.2. Many hardness results for the general setting actually apply here as well. Our main contribution are pseudo-polynomial-time algorithms for a certain subclass of two-terminal series-parallel graphs, which we call *sps-graphs*.

Outline. We first argue why only two of the four demand types defined in Section 2.1 are relevant to the expansion problems (Section 3.2). Then, we elaborate

on the complexity of finding feasible expansions in Section 3.3. The following sections include new hardness results (Section 3.4) and polynomial-time algorithms (Section 3.5). In Section 3.6 we consider networks with only one source and one sink. Finally, we conclude this chapter with a short summary and an outlook over further research directions in Section 3.7

3.2 Relations Between the Demand Types

In Section 2.1 we define networks with four types of demands: adjustable, strict, basic, and fixed. As a first step, we study relations between networks of different types in the context of the expansion planning problems in this chapter. First, we show that adjustable and strict demands are equivalent.

Lemma 3.1. *Let $\mathcal{N}$ be an instance of* Flow Expansion *or* Electrical Flow Expansion *with adjustable demands. Then, there is a network with strict demands that is equivalent to $\mathcal{N}$.*

Proof. Suppose there is a vertex v in $\mathcal{N}$ with a demand interval $D(v) = [l, u]$ that contains both positive and negative values. That is $l < 0$ and $u > 0$. We call such a vertex *hybrid*. The key insight is that we can write $[l, u] = [l, 0] + [0, u]$. Neither interval on the right contains both positive and negative values. In this spirit we add two new vertices s_v and t_v with edges to v. The vertex s_v gets a demand interval of $[l, 0]$, and t_v gets $[0, u]$, whereas the demand interval of v is set to $\{0\}$. The additional edges vs_v and vt_v belong to the set of existing edges, and their capacities are set to $-l$ and u, respectively. In the case of an electrical network, we also need to define their susceptances. However, the two edges do not lie on any cycle such that their susceptances do not matter. Thus, we may set them to any value, e.g., 1.

We apply this construction to all hybrid vertices of $\mathcal{N}$ and obtain $\mathcal{N}'$. There is a one-to-one correspondence between expansions of $\mathcal{N}$ and expansion of $\mathcal{N}'$ by choosing the same set of candidate edges. Consider an expansion $\mathcal{H}$ of $\mathcal{N}$ and the corresponding expansion $\mathcal{H}'$ of $\mathcal{N}'$. It is easy to see that every (electrical) flow f in $\mathcal{H}$ can be extended to an (electrical) flow f' in $\mathcal{H}'$ by setting $f'(vs_v) = \min\{c_f(v), 0\}$ and $f'(vt_v) = \max\{c_f(v), 0\}$. Conversely, the restriction of an (electrical) flow f' in $\mathcal{H}'$ to the edges of $\mathcal{N}$ yield an (electrical) flow in $\mathcal{H}$. Hence, $\mathcal{H}$ is feasible if and only if $\mathcal{H}'$ is feasible. □

Second, we observe that basic networks always admit a flow that is 0 everywhere. Hence, the expansion that contains only the existing edges is feasible and has costs 0. Since the candidate edges have non-negative costs, this is as cheap as possible.

Observation 3.2. Let $\mathcal{N}$ be an instance of Flow Expansion or Electrical Flow Expansion on the graph $(V, E_{\text{ex}} \cup E_{\text{cand}})$ with basic demands. Then, (V, E_{ex}) is an optimal expansion, and $f \equiv 0$ is a feasible (electrical) flow in this expansion.

These two results allow us to restrict our attention to two types of demands in this chapter: fixed and adjustable.

3.3 Finding Feasible Expansions

Before trying to find optimal expansions, we analyze how difficult it is to find any feasible expansion. Flow Expansion is based on a graph-theoretical flow model. Hence, any flow f in any feasible expansion $\mathcal{H}$ of a network $\mathcal{N}$ can be extended to a flow f' in the whole network $\mathcal{N}$ by setting $f'(e) = 0$ for all edges $e \in E(\mathcal{N}) \setminus E(\mathcal{H})$. Thus, there is a feasible expansion of $\mathcal{N}$ if and only if there is a flow in $\mathcal{N}$. Checking for a flow in $\mathcal{N}$ is possible in polynomial time; see Section 2.1.1.

Proposition 3.3. *It can be determined in polynomial time whether an instance of Flow Expansion has a feasible expansion.*

In contrast, extending flows like this does not work for electrical flows because the angle difference between the endpoints of an edge $e \in E(\mathcal{N}) \setminus E(\mathcal{H})$ may require a non-zero flow on e. In fact, it is known that finding feasible solutions to the Maximum Transmission Switching Flow problem is $\mathcal{NP}$-hard in certain cases [LGV14, Koc+16]. An instance of Maximum Transmission Switching Flow consists of an electrical flow network $\mathcal{N}$. The goal is to find a suitable subnetwork $\mathcal{N}'$ of $\mathcal{N}$ and an electrical flow in $\mathcal{N}'$ such that the total flow value is maximized. Since Maximum Transmission Switching Flow and Electrical Flow Expansion are closely related, this hardness can be transferred to Electrical Flow Expansion.

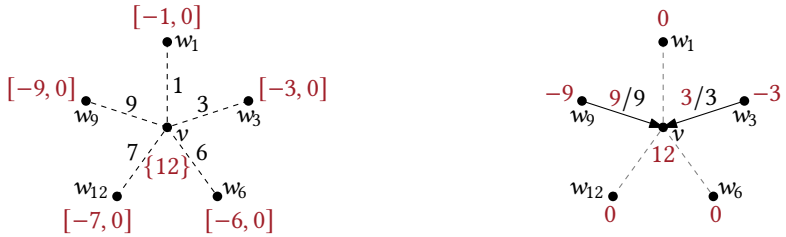
Proposition 3.4. *The problem of determining whether an instance of Electrical Flow Expansion has a feasible expansion is*

(a) $\mathcal{NP}$-complete if the underlying graph is a cactus and the demands are adjustable,

(b) $\mathcal{NP}$-complete if the underlying graph is a partial 2-tree and the demands are fixed,

(c) strongly $\mathcal{NP}$-complete if the underlying graph is planar and the demands are fixed.

Proof. The feasibility variant of Maximum Transmission Switching Flow is equivalent to Electrical Flow Expansion when the existing network contains no edges.



(a) The network. **(b)** The feasible expansion of cost 12.

Figure 3.1: An example of the network constructed in the proof of Theorem 3.5. The network corresponds to the instance of Subset Sum defined by $A = \{1, 3, 6, 7, 9\}$ and $k = 12$.

That is, all edges are candidate edges. Hence, the hardness results for the Maximum Transmission Switching Flow problem can be transferred directly to Electrical Flow Expansion. More precisely, Items (a) and (c) in this proposition are derived from [LGV14, Thm. 1], and Item (b) is derived from [Koc+16, Thm. 3.1]. □

3.4 Hardness

There are polynomial-time algorithms both for computing graph-theoretical flows and for computing electrical flows; see Section 2.1. Hence, it can be checked if any given expansion of a network is feasible in polynomial time. This implies that determining whether a network has a feasible expansion lies in $\mathcal{NP}$. Moreover, the costs of an expansion can be computed in linear time. Thus, the decision variants of both Flow Expansion and Electrical Flow Expansion are in $\mathcal{NP}$ as well. We start by proving that determining whether a network with adjustable demands has a feasible expansion is $\mathcal{NP}$-complete.

Theorem 3.5. *The decision variant of Flow Expansion with adjustable demands and a star as underlying graph is $\mathcal{NP}$-complete.*

Proof. We prove the hardness by a reduction from Subset Sum. An instance of Subset Sum consists of a set $A \subseteq \mathbb{N}$ and a natural number $k \in \mathbb{N}$. Such an instance (A, k) belongs to Subset Sum if and only if there is a subset of A whose elements sum to k.

We construct a star graph $K_{1,|A|}$ with center v and leaves w_a for $a \in A$; see Figure 3.1 for an example. All edges are candidate edges. The center is a sink with demand interval $D(v) = \{k\}$. For $a \in A$, we set $D(w_a) = [-a, 0]$, $\mathrm{cap}(v w_a) = a$, and

$c(v\,w_a) = a$. We call the resulting network $\mathcal{N}$. It can clearly be computed in linear time. We further claim that there is an expansion of $\mathcal{N}$ with cost at most k if and only if (A, k) belongs to Subset Sum.

By construction there is a one-to-one correspondence between elements of A and leaves (and thus edges) of $K_{1,|A|}$. Hence, there is a bijection Φ between subsets of A and expansions of $\mathcal{N}$, which are determined by subsets of edges. Consider any $A' \subseteq A$, and let k' be the sum of elements of A'. We further observe that in $\Phi(A')$ the sum of the edge capacities is k'. Therefore, there is no flow in $\Phi(A')$ if $k' < k$ since the demand of the center u cannot be satisfied. However, if $k' \geq k$, the definition of the demands of the leaves as $D(w_a) = [-\mathrm{cap}(v\,w_a), 0]$ implies that there is a flow in $\Phi(A')$. In total, $\Phi(A')$ is feasible if and only if $k' \geq k$. Since $\mathrm{cost}(\Phi(A')) = k'$, the total cost of the expansion is at most k if and only if $k' \leq k$. This shows that feasible expansions of cost at most k correspond to subsets $A' \subseteq A$ with sum k and vice versa. □

Note that on trees every graph-theoretical flow is an electrical flow. Hence, the reduction proves the hardness of Electrical Flow Expansion on trees as well.

Corollary 3.6. *The decision variant of Electrical Flow Expansion with adjustable demands and a tree as underlying graph is $\mathcal{NP}$-complete.*

In the reduction above we crucially use the fact that the consumption at the leaves are not fixed. Selecting which of the leaves produce flow and which do not is the key choice when searching for a cheap feasible expansion. When we restrict ourselves to networks with fixed demands, this construction does not work anymore, at least not directly. However, we do know the total consumption in any flow in any feasible expansion, namely the demand of the center. Due to the simple structure of the network and with carefully choosing the susceptances, the standard approach of adding a super source to the network gives an equivalent network. Hence, we obtain the $\mathcal{NP}$-hardness of Electrical Flow Expansion with fixed demands.

Theorem 3.7. *The decision variant of Electrical Flow Expansion with fixed demands is $\mathcal{NP}$-complete even if the underlying network is a $K_{2,n}$ for $n \in \mathbb{N}$.*

Proof. As in Theorem 3.5 we reduce an instance (A, k) of Subset Sum to Electrical Flow Expansion. This time, however, the demands are fixed. We have one source s with a demand of $-k$ and one sink t with a demand of k. For every $a \in A$ there is a path sv_at of candidate edges with costs a, susceptances a, and capacities a. Let the resulting network be $\mathcal{N}$. The vertices v_a have a demand of 0. We claim that there is an feasible expansion $\mathcal{H}$ of $\mathcal{N}$ with $\mathrm{cost}(\mathcal{H}) \leq 2k$ if and only if the instance (A, k) has a solution.

If only one edge of any path sv_at is included in an expansion $\mathcal{H}$, the flow on this edge has to be 0 in any flow. Hence, leaving this edge out is both cheaper and allows for the same flows. Therefore, we may treat the candidate edges on one path as a single unit.

Every subset A' of A corresponds to an expansion $\mathcal{H}_{A'}$ of $\mathcal{N}$, where the edges sv_a and v_at are included in the expansion if and only if $a \in A'$. It is easy to verify that

$$\operatorname{cost}(\mathcal{H}_{A'}) = 2 \sum_{a \in A'} a.$$

Let k' be the sum of the elements in A', i.e., $k' = \operatorname{cost}(\mathcal{H}_{A'})/2$. The edges of $\mathcal{H}_{A'}$ incident to the source s form a cut with capacity k'. Hence, if $k' < k$, there is no flow in $\mathcal{H}_{A'}$. Otherwise, if $k' \geq k$, setting $\theta(s) = 2k/k'$, $\theta(v_a) = k/k'$ for all $a \in A$, and $\theta(t) = 0$ induces an electrical flow in $\mathcal{H}_{A'}$.

In total, if $A' \subseteq A$ is a solution of (A, k), then $\mathcal{H}_{A'}$ is a feasible expansion of $\mathcal{N}$ with $\operatorname{cost}(\mathcal{H}_{A'}) = 2k$. Conversely, $\mathcal{H}$ is a feasible expansion of $\mathcal{N}$ with $\operatorname{cost}(\mathcal{H}) \leq 2k$, then we may assume that $\mathcal{H} = \mathcal{H}_{A'}$ for some $A' \subseteq A$. Let k' be the sum of A'. The feasibility of $\mathcal{H}$ yields $k' \geq k$, and the total cost bound yields $k' \leq k$. Thus, A' is a solution of (A, k). □

Leaving out the susceptances in the reduction above directly shows the $\mathcal{NP}$-hardness of Flow Expansion with fixed demands as well.

Corollary 3.8. *The decision variant of Flow Expansion with fixed demands is $\mathcal{NP}$-complete even if the underlying network is a $K_{2,n}$ for $n \in \mathbb{N}$.*

The problem Steiner Tree can be reduced to both Flow Expansion and Electrical Flow Expansion [MPS10]. An instance of Steiner Tree consists of a weighted undirected graph $G = (V, E)$ and a set of terminals $T \subseteq V$. The goal is to find the minimum-weight subgraph of G that connects all terminals. While not noted explicitly, the reduction also implies that the approximation hardness of Steiner Tree can be transferred as well. More precisely, it is known that it is $\mathcal{NP}$-hard to approximate Steiner Tree within a factor of $96/95$ [CC08]. Thus, we obtain a similar result for Flow Expansion and Electrical Flow Expansion.

Proposition 3.9. *It is $\mathcal{NP}$-hard to approximate Flow Expansion and Electrical Flow Expansion within a factor of $96/95$ even if the demands are fixed and there is only one source.*

3.5 Polynomial-Time Algorithms for Networks with Fixed Demands

In this section we only consider networks with fixed demands. That is, their demand intervals contain only one value, and we can replace the demand intervals by demands, i.e., we have a function d that assigns the demand value $d(v)$ to each vertex v.

Lemma 3.10. *In networks with fixed demands, it can be determined in linear time, which bridges are part of an optimal expansion. Moreover, the amounts of flow on these bridges are the same for all flows in all expansions, and the amounts can be determined in linear time as well.*

Proof. Let vw be a bridge in the underlying graph G of the network $\mathcal{N}$. By definition, the removal of vw disconnects G into two components G_v and G_w that contain v and w, respectively. Let s_v and s_w be the sums of the demands at vertices of G_v and G_w, respectively. If $s_v \neq -s_w$, then there is no feasible expansion. Otherwise, we have $f(vw) = s_w$ for any flow f in any expansion. Hence, if $s_w \neq 0$, any feasible expansion contains vw. In contrast, if $s_w = 0$ and vw is a candidate, the edge vw may be removed from any expansion without affecting the feasibility of the expansion. Moreover, doing so may only reduce the costs.

Since this argument holds for every bridge, we obtain that the amount of flow on the bridges is the same in any flow in any expansion of $\mathcal{N}$. Additionally, if $\mathcal{N}$ has a feasible expansion, then there is an optimal expansion that contains only those bridges for which the demand sums defined above are not 0.

It remains to show how to compute the demand sums and this set of bridges in linear time. We first check whether the sum of the demands of all vertices is 0. If not, there is no feasible expansion, and we stop. Then, we determine the bridges of G in linear time using a modified depth-first search (DFS) [HT73]. We then root the resulting DFS-tree T at an arbitrary vertex and perform a bottom-up traversal to compute the demand sums, which takes linear time. Afterwards, every vertex v is annotated with the sums a_v of the demands at the vertices in the subtree below v. Every spanning tree, and in particular T, contains every bridge. Let vw be a bridge. We may assume without loss of generality that w is a child of v in T. Then, a_w is the sum of the demands in the component of $G - vw$ that contains w. By the argument above, the amount of flow on vw is a_w in every flow. Furthermore, we can determine whether to include vw in an expansion (if $a_v \neq 0$ or vw is an existing edge) or not. □

In a first step we apply this lemma to trees. There, every edge is a bridge. Hence, the algorithm in the proof of Lemma 3.10 yields a unique expansion, and either this expansion is optimal or there is no feasible expansion. Moreover, we obtain an assignment of values to the edges that satisfies the flow conservation constraints at

the vertices. To determine whether it is a flow, it remains to check if the assigned values exceed the capacities. This is clearly possible in linear time.

Proposition 3.11. *On trees both* Flow Expansion *with fixed demands and* Electrical Flow Expansion *with fixed demands can be solved optimally in $\mathcal{O}(n)$ time.*

Having understood the case expansion problem in graphs where every edge is a bridge, we move on to a slightly more complex graph class: cacti. Since no two cycles share an edge in a cactus, all biconnected components are either a bridge or a cycle.

We assume in the remainder of this section that the sum of all demands is 0. Otherwise, there is no feasible expansion. This prerequisite can be checked easily in linear time.

We first argue that the biconnected components can be considered independently. Two biconnected components have at most one vertex in common. A vertex that belongs to two biconnected components is a *cut-vertex*. Consider a cut-vertex v. The graph $G - v$ consists of $k \geq 2$ components $G_1, \ldots, G_k$. We define $H_i = G_i + v$ for $i \in \{1, \ldots, k\}$. There is a unique way of distributing the demand of v into demands for each graph H_i such that the total demand in the subnetwork $\mathcal{N}_i$ induced by H_i is 0. We can then consider the networks $\mathcal{N}_1, \ldots, \mathcal{N}_k$ independently. Repeating this procedure for all cut-vertices, we decompose the network into subnetworks induced by the biconnected components. For an efficient calculation of the subnetworks for the biconnected components, we first compute the block-cut tree in linear time [HT73]. The demands of the cut-vertices in the subnetworks can be computed by traversing the block-cut tree bottom-up. All leaves of the block-cut tree are biconnected components that contain exactly one cut-vertex. For this vertex the demand can easily be determined. Once all children of a node corresponding to a biconnected component have been handled, the demands of all but (at most) one cut-vertex have been fixed. Hence, we can proceed as for the leaf. In total, this procedure can be implemented to run in linear time.

Lemma 3.12. *The biconnected components can be considered independently and the subnetworks for the biconnected components can be computed in linear time.*

As mentioned above the biconnected components of a cactus are either bridges or cycles. The bridges can be handled by Lemma 3.10. Thus, it remains to show how the cycles can be handled efficiently.

Lemma 3.13. *In a cycle C an optimal expansion can be computed in $\mathcal{O}(|C|^2)$ time.*

Proof. Let $\mathcal{N}$ be a network on a cycle C. An optimal expansion (if it exists) either includes all edges or at least one of the edges is missing. Thus, there are $|C| + 1$ cases: one case if all edges are present, and $C - e$ for every edge e. In the first case, we

simply check whether there is an (electrical) flow in the network induced by C. We claim that this is possible in $\mathcal{O}(|C|^2)$ time. For graph-theoretical flows an algorithm for the general case has this running time [Orl13]. For electrical flows we can reduce this to at most $|C| - 1$ computations of electrical flows between two vertices, each taking $\mathcal{O}(|C|)$ time; see Section 2.1.2. The cost of this expansion is the cost of all candidate edges on C. In the other cases, $C - e$ is a path. Hence, there is at most one flow, which can be checked in $\mathcal{O}(|C|)$ time, e.g., by applying Lemma 3.10. Note that the flow may be 0 on some edges. If these edges are candidate edges, we do not need to include them in the expansion, and their costs can be ignored. Finally, we determine which of these expansions is the cheapest feasible one, and we return it (or that there is no feasible expansion). In total, this takes $\mathcal{O}(|C|^2)$ time. □

All that remains for an algorithm for the expansion problems on cacti is to put everything together.

Theorem 3.14. *On cacti both* Flow Expansion *with fixed demands and* Electrical Flow Expansion *with fixed demands can be solved optimally in $\mathcal{O}(n^2)$ time.*

Proof. We first decompose the network $\mathcal{N}$ into the subnetworks induced by the biconnected components in linear time (Lemma 3.12). These are either single edges (and thus bridges), which can be handled in linear time in total by Lemma 3.10, or cycles, which can be handled in time quadratic in the number of edges on the cycle by Lemma 3.13. Afterwards, we have an optimal expansion in each of the subnetworks induced by the biconnected components, or we determine that there is no feasible expansion in some subnetwork. In the latter case, we conclude that there is no feasible expansion of the whole network. In the former case, we combine the optimal expansions of the subnetwork to an expansion $\mathcal{H}$ of the network $\mathcal{N}$. Clearly, $\mathcal{H}$ is feasible (by Lemma 3.12) and has minimum costs. Otherwise, it would contain a cheaper expansion for one of the subnetworks.

The procedure takes $\mathcal{O}(n)$ time for decomposing the network and to combine the expansions. The optimal expansion of a biconnected component with ℓ edges can be computed in $\mathcal{O}(\ell^2)$ time by Lemmas 3.10 and 3.13. Hence, the total running time lies in $\mathcal{O}(n^2)$. □

3.6 Single Source, Single Sink

In this section we consider *two-terminal networks*, which are networks where for all but two vertices the demand intervals are $\{0\}$. Let these two vertices be s and t. If both $D(s)$ and $D(t)$ contain 0, the function that is 0 on all edges defines an (electrical) flow in any expansion. Thus, adding no candidate edges yields an optimal expansion.

Therefore, we may assume that at least one of the intervals does not contain 0, but only, say, positive values. In any flow f in any expansion, we have $c_f(s) = -c_f(t)$. Hence, we can ignore the positive values in the demand interval of the other vertex, and we may assume without loss of generality that $D(s)$ contains only negative values and $D(t)$ contains only positive values. Then, s is the unique source and t is the unique sink.

Lemma 3.15. *Let $\mathcal{N}$ be an (electrical) two-terminal network with source s and sink t. Let $a = \max\{-\max D(s), \min D(t)\}$, and let $\mathcal{H}$ be an expansion of $\mathcal{N}$. If there is an (electrical) flow in $\mathcal{H}$, then there is an (electrical) flow f in $\mathcal{H}$ such that $c_f(s) = -a$ and $c_f(t) = a$.*

Proof. Let f' be an (electrical) flow in $\mathcal{H}$. We define $\alpha = a/c_{f'}(t)$. By the definition of a and the fact that $c_{f'}(s) = -c_{f'}(t)$, it holds that $\alpha \leq 1$. Hence, $f := \alpha \cdot f'$ is an (electrical) flow in $\mathcal{H}$ with the desired consumptions at s and t. □

This lemma allows us to treat the demands as fixed with $d(s) = -a$, $d(t) = a$, and $d(v) = 0$ for all other vertices v.

In Theorem 3.7 and Corollary 3.8 we already establish that Flow Expansion and Electrical Flow Expansion are $\mathcal{NP}$-complete on the class of parallel paths between a single source and a single sink. In the remainder of this section we present two pseudo-polynomial-time algorithms for (Electrical) Flow Expansion on graph classes that include (among others) parallel paths. We start with an algorithm for Flow Expansion on series-parallel graphs.

Proposition 3.16. *Let $\mathcal{N}$ be an instance of Flow Expansion on a two-terminal network with integral costs and capacities, where the underlying graph G is two-terminal series-parallel and the terminals of G are the source and the sink of $\mathcal{N}$. An optimal expansion can be computed in $\mathcal{O}(n^3 \cdot \min\{C^2, U^2\})$ time, where the candidate costs and capacities are bounded by C and U, respectively.*

Proof. We only describe an algorithm that obtains a running time of $\mathcal{O}(n^3C^2)$ in detail. At the end we sketch how to modify it to get an algorithm with a running time of $\mathcal{O}(n^3U^2)$. Choosing the algorithm with the better running time for each instance then gives an algorithm with the claimed running time.

Let G be a series-parallel graph with n vertices and m edges. Its structure can be represented by an sp-tree $\mathcal{T}$ as described in Section 2.3. Each internal node α of $\mathcal{T}$ represents a subgraph G_α of G. We compute an optimal solution by dynamic programming on $\mathcal{T}$.

For every node α in the tree we compute the function $F_\alpha \colon \mathbb{N}_0 \to \mathbb{N}_0$, where $F_\alpha(a)$ represents the maximum total flow value in G_α when the costs of the selected candidate edges is at most k. Recall that we assumed all input values to be integral.

Hence, it is sufficient for F_α to be a map from the natural numbers. Let ρ be the root of the tree. If we know F_ρ, we are able to compute the minimum cost for an expansion as follows. The total cost of all candidate edges is at most mC. Hence, if $F_\rho(mC) < d(t)$, then there is no feasible expansion of $\mathcal{N}$. Otherwise, we find the smallest $k \in \mathbb{N}_0$ such that $F_\rho(k) \geq d(t)$.

It remains to show how F_α for all nodes α can be computed efficiently. Note that if G_α has m' edges, then $F_\alpha(k) = F_\alpha(m'C)$ for all $k \geq m'C$. Hence, we only need to compute and store $F_\alpha(0), \ldots, F_\alpha(m'C)$.

At a leaf α, which represents an edge e, we have

$$F_\alpha(k) = \begin{cases} \operatorname{cap}(e), & e \in E_{\text{ex}}, \\ 0, & e \in E_{\text{cand}} \text{ and } k < c(e), \\ \operatorname{cap}(e), & e \in E_{\text{cand}} \text{ and } k \geq c(e). \end{cases}$$

If α is an internal node it has two children β and γ. If the subgraph G_α is formed by a series composition, it holds that

$$F_\alpha(k) = \max\{\min\{F_\beta(k_1), F_\gamma(k_2)\} \mid k_1, k_2 \in \mathbb{N}_0, k_1 + k_2 = k\}.$$

If G_α is formed by a parallel composition, we have

$$F_\alpha(k) = \max\{F_\beta(k_1) + F_\gamma(k_2) \mid k_1, k_2 \in \mathbb{N}_0, k_1 + k_2 = k\}.$$

Every function value can be computed in $\mathcal{O}(mC) = \mathcal{O}(nC)$ time. Since we compute $\mathcal{O}(mC)$ values for each of the $\mathcal{O}(n)$ functions, we have a total running time of $\mathcal{O}(n^3C^2)$.

A similar approach gives a running time of $\mathcal{O}(n^3U^2)$. We compute at each node α of the sp-tree the function F'_α where $F'_\alpha(k')$ is the minimum cost of an expansion that allows for a flow of k' through the subgraph G_α represented by α. Note that by Corollary 2.2 there is an integral flow in the subgraph G_α. Thus, it suffices to only consider $F'_\alpha(k')$ for $k' \in \mathbb{N}_0$. An analysis analogous to the one above shows that we compute $\mathcal{O}(nU)$ function values for $\mathcal{O}(n)$ functions, and each computation takes $\mathcal{O}(nU)$ time. In total, this gives a running time in $\mathcal{O}(n^3U^2)$. □

Note that the algorithm in the proof above cannot easily be adapted to work on electrical flow networks. It builds on the fact that two parallel subnetworks do not interact with each other, except that their out-flows at the sinks (and in-flows at the sources) add. In contrast to that, if we considered an electrical flow, the angle differences between the source and sink of the two subnetworks would have to be the same for both subnetworks. This additional influence cannot be captured as easily.

Moreover, with electrical flows it is not possible anymore to assume that the resulting flows are integral, which further complicates the situation.

In fact, the analogous problem for electrical flows, Electrical Flow Expansion, is $\mathcal{NP}$-hard on partial 2-trees even if all capacities are 1 and edge costs are ignored (but the susceptances are not constant) [Gra+18]. This result does not preclude the existence of a pseudo-polynomial-time algorithm. However, assuming $\mathcal{P} \neq \mathcal{NP}$, the running time of such a hypothetical algorithm would need to depend on the magnitude of the susceptances as well. We leave finding such an algorithm as an open question.

But we prove that there is a pseudo-polynomial-time algorithm for Electrical Flow Expansion (and for Flow Expansion) on a subclass of the series-parallel graph, namely those graphs that are formed by a series composition of blocks consisting of parallel paths. We call such graphs *sps-graphs* (for series–parallel–series).

Proposition 3.17. *Let $\mathcal{N}$ be an instance of Electrical Flow Expansion on a two-terminal network with integral costs bounded by C, where the underlying graph G is an sps-graph and the terminals of G are the source and the sink of $\mathcal{N}$. An optimal expansion can be computed in $\mathcal{O}(n^3 CT)$ time, where T is the time it takes to add numbers with $\mathcal{O}(n^2\|b\| + n\|\mathrm{cap}\|)$ bits, where $\|b\|$ and $\|\mathrm{cap}\|$ are the maximum number of bits needed to represent a value of b and* cap.

Proof. We note that in sps-graphs each block of parallel paths forms a biconnected component. Thus, it can be considered independently by Lemma 3.12. We reduce finding an optimal solution Flow Expansion on parallel paths to solving a linear number of Minimum Knapsack instances. An instance of Minimum Knapsack consists of a set of objects A, some $k \in \mathbb{N}_0$, as well as a cost $c(a) \in \mathbb{N}_0$ and a value $v(a) \in \mathbb{N}_0$ for each $a \in A$. The goal is to find a subset $A' \subseteq A$ with minimum cost such that the sum of the values is at least k. A straightforward dynamic programming approach yields a $\mathcal{O}(n^2 \max_{a \in A} c(a))$-time algorithm for Minimum Knapsack [KPP04, Sec. 13.3.3 and Thm. 7.2.1].

Let $P_1, \ldots, P_k$ be the paths between the source s and the sink t. We denote their edge sets by $E(P_i)$. We associate each path P_i with its costs $c(P_i)$, susceptance $b(P_i)$, and capacity $\mathrm{cap}(P_i)$, which are defined by

$$
\begin{aligned}
c(P_i) &= \sum_{e \in E(P_i) \cap E_{\mathrm{cand}}} c(e), \\
b(P_i) &= \frac{1}{\sum_{e \in E(P_i)} \frac{1}{b(e)}}, \\
\mathrm{cap}(P_i) &= \min\{\mathrm{cap}(e) \mid e \in E(P_i)\}.
\end{aligned}
$$

We observe that $b(P_i)$ can be represented with $\mathcal{O}(n\|b\|)$ bits. We call the paths with candidate edges *candidate paths* and the others *existing paths*. Furthermore, we denote the set of existing paths by $\mathcal{Q}_{\text{ex}}$ and the set of candidate paths by $\mathcal{Q}_{\text{cand}}$.

The key idea is to reduce the selection of the paths to a series of Minimum Knapsack computations. Every path P_i that is contained in an expansion has cost $c(P_i)$, and naively one may assume that it can transport $\text{cap}(P_i)$ units of flow. However, by Kirchhoff's voltage law, the flows on the paths cannot be considered independently. For example, sending $\text{cap}(P_i)$ units along P_i may induce an angle difference between s and t of $\Phi(P_i)$. That is, if $\mathcal{H}$ is an expansion that completely contains P_i and f is an electrical flow in $\mathcal{H}$, then the angle difference between s and t is at most $\Phi(P_i)$. More concretely, the maximum angle difference $\Phi(P_i)$ induced by P_i is given by $\Phi(P_i) = \text{cap}(P_i)/b(P_i)$, which can be represented with $\mathcal{O}(n\|b\|+\|\text{cap}\|)$ bits. However, another path P_j may induce an angle difference of $\Phi(P_j)$, which may differ from $\Phi(P_i)$. Having both paths in an expansion, thus restricts the angle difference to $\min\{\Phi(P_i), \Phi(P_j)\}$. This in turn restricts the maximum flow values on one path to possibly less than the capacities.

But every expansion $\mathcal{H}$ completely contains a path P_j with minimum value $\Phi(P_j)$ among all paths in $\mathcal{H}$. We may then assume that the angle difference between s and t is exactly $\Phi(P_j)$, which allows us to assign values $F(P_i)$ to all paths P_i that describe the amount of flow along P_i if the angle difference is $\Phi(P_j)$. To simplify the notation we assume that the paths are ordered such that $\Phi(P_i) \leq \Phi(P_j)$ for $i \leq j$. Then, P_j has the minimum index of all paths selected for $\mathcal{H}$.

For each choice of $P_j \in \mathcal{Q}_{\text{ex}} \cup \mathcal{Q}_{\text{cand}}$, we reduce the search for a cost-minimal expansion to an instance of Minimum Knapsack. For each path P_i with $i \geq j$ the value $F(P_i)$ is given by $F(P_i) = b(P_i) \cdot \Phi(P_j)$, which can be represented in $\mathcal{O}(n\|b\| + \|\text{cap}\|)$ bits. These represent the flow values we can achieve if we select P_i. It remains to compute how much flow we actually need. Part of the demand $d(t)$ may already be satisfied via existing paths. With the fixed angle difference of $\Phi(P_j)$, the remaining demand is

$$d_0 = d(t) - \sum_{P_i \in \mathcal{Q}_{\text{ex}}} F(P_i).$$

The goal is then to find a subset $\mathcal{Q}' \subseteq \{P_i \in \mathcal{Q}_{\text{cand}} \mid i \geq j\}$ such that $\sum_{P \in \mathcal{Q}'} F(P) \geq d_0$ and $\sum_{P \in \mathcal{Q}'} c(P)$ is minimal. This would be an instance of the Minimum Knapsack problems if all values were integral. However, the values $F(P_i)$ may be non-integral. Instead of working with these values, we define α as the smallest integer such that $\alpha d_0 \in \mathbb{N}_0$ and $\alpha \cdot F(P_i) \in \mathbb{N}_0$ for all $i \geq j$. Let then $F'(P_i) = \alpha F(P_i)$ and $d_0' = \alpha d_0$. The value α is bounded by the product of the denominators of all $F(P_i)$. Hence, all new values can be represented in $\mathcal{O}(n^2\|b\| + n\|\text{cap}\|)$ bits. The resulting instance of the Minimum Knapsack problem can hence be solved in $\mathcal{O}(n^2 CT)$ time [KPP04].

However, initially we do not know the path P_j with the minimum index in an optimal expansion. We therefore try all values of j between 1 and $\min\{i \mid P_i \in \mathcal{Q}_{\text{ex}}\}$. The cost-minimal expansion then corresponds to the cheapest solution of one of these Minimum Knapsack instances. This adds another factor of $\mathcal{O}(n)$ to the overall running time. □

3.7 Conclusion

Flow Expansion and Electrical Flow Expansion are optimization problems that can be seen as variants of the Minimum Cost Flow problem. Their corresponding decision problems are $\mathcal{NP}$-complete even in very simple cases, e.g., if the network is a star (Theorem 3.5 and Corollary 3.6) or if there are only parallel paths between the only source and the only sink (Theorem 3.7 and Corollary 3.8). For some restricted cases, there are (pseudo-)polynomial-time algorithms.

Our results show that the graph classes and the demand type (fixed vs. adjustable demand) have a crucial impact on the complexity of the problems. Interestingly, our results do not show a strong difference between the flow models, i.e., between Flow Expansion and Electrical Flow Expansion. The only notable difference is Proposition 3.16, whose proof is based on a dynamic program on the structure of a partial 2-tree. It is not clear how to adapt the proof to electrical flows. A straightforward follow-up questions is therefore: Can the algorithm be adapted to work with electrical flows as well (in the same running time)? Or more general, are there cases where the complexities of Flow Expansion and Electrical Flow Expansion clearly differ? As another possible research direction it may also be possible to extend the results for partial 2-trees to graphs of bounded treewidth.

4 Expanding Electrical Networks to Prevent Critical Edges

In the basic expansion planning formulation that we analyze in Chapter 3 no redundancy is required in case some transmission equipment fails. But in practice it is important to ensure that the transmission network is reliable. In particular, it should remain operable if one piece of equipment fails ($N-1$ criterion). In this chapter we consider a criticality measure by Witthaut et al. [Wit+16], We observe that networks without critical edges tend to satisfy the $N-1$ criterion even though (as we show) there are networks that satisfy either criterion but not the other.

We formulate the criticality measure as a set of linear constraints, which may form a building block in network design problems. To exemplify this usage, we introduce these constraints into a generalization of Electrical Flow Expansion to multiple timestamps, obtaining MILPs for two expansion planning problems. We study the effects of adding these constraints on the time needed for solving the models. Furthermore, we present a simple heuristic for one of the two problems, which often finds optimal solutions but in less time than solving the MILP.

This chapter is based on joint work with Dorothea Wagner [WW20].

4.1 Introduction

When designing a new electric transmission grid or re-designing an existing one, one strives to make the grid robust against potential equipment failures. Otherwise, a single failure may cause a collapse of large parts of the grid leading to a widespread black out. To prevent such widespread failures, one therefore has to take these

equipment failures into account both during the design and the operation phase. In this work we focus on the design phase.

A widely used reliability criterion is the $N-1$ *criterion* [ZA05]. It roughly states that the grid must be operable even if a single piece of equipment (e.g., a transmission line or a generator) fails. It is widely used both while designing [SMB10, CMT07, LHS19] and operating transmission and distribution grids [Mey+18]. This criterion may also be extended to include the failure of up to k pieces of equipment. This is then called the $N-k$ *criterion* [MSA15].

Typically, these criteria consider the static behavior of the grid after the failure. They do not include the dynamic behavior of, e.g., a line failure. In contrast, to assess the severity of edge failures one may need to take the dynamic behavior of the voltage angles in transmission grids into account as well [Wit+16]. Lines are classified as either *critical* if their failure causes a widespread outage or *non-critical* if the failure still leaves the grid operable. Two simple measures that try to predict for each line whether they are critical or not were developed [Wit+16]. The first measure bases the decision on the maximum graph-theoretical flow between the endpoints of the line in the residual network after the line is removed; see Section 4.3 for a formal definition. If the fraction of the power flow on the line and the value of the maximum is larger than some threshold h, the line is predicted to be critical. The second measure is based on the linear response on small perturbations in the existing power flow. The authors claim that the two measures more accurately predict critical edges than standard load flow analyses. In this chapter, we present how to incorporate the first measure into transmission network design problems, e.g., Electrical Flow Expansion.

Related Work. There is a large body of research on Transmission Network Expansion Planning. For a general overview we refer to Chapter 1 and the book by Lumbreras, Abdi, and Ramos [LAR21]. In this section, we focus on the works that incorporate reliability criteria such as the $N-1$ criterion [CMT07, SMB10].

Transmission Network Expansion Planning including both AC and DC links considering the $N-1$ criterion may be formulated as an MILP [DEG17]. Moreover, that work contains a method to reduce the search space of the MILP formulation, which significantly decreases the time to solve the model.

Choi, Mount, and Thomas [CMT07] formulate an integer programming formulation for Transmission Network Expansion Planning taking the $N-k$ criterion and variations thereof into account. This means, they consider contingencies with up to k component failures. However, to reduce the complexity of the model, they ignore contingencies with probabilities below a given threshold (10^{-9} in their study). Moreira, Street, and Arroyo [MSA15] address the computational complexity of explicitly modeling the $N-k$ criterion differently. They formulate this problem as a trilevel

mixed-integer program and solve it using Benders decomposition.

Instead of employing the deterministic $N-1$ criterion, Shortle, Rebennack, and Glover [SRG14] consider the stochastic nature of blackouts. They present a variant of the transmission expansion planning problem with the goal of minimizing the probability of blackouts.

Reliability criteria are not only important for designing but also for operating transmission grids. A relaxed version of the $N-1$ criterion is proposed by Zima and Andersson [ZA05]. They find that using this relaxed criterion during dispatch reduces the expected blackout size. Heylen et al. [Hey+19] compare six reliability criteria including the $N-1$ criterion and various probabilistic criteria. Their case study shows that probabilistic criteria allow for lower expected total cost of reliability management.

Contribution and Outline. We present a linear program formulation of the criticality criterion by Witthaut et al. [Wit+16]. This formulation is general in the sense that it can be included in various transmission network design problems. As an example application, we include it in a basic version of the Transmission Network Expansion Planning problem. This yields the two problems: Criticality-Constrained Transmission Network Expansion Planning (CC-TNEP) and Criticality Minimal Expansion (CME). In the former problem, the total criticality of all edges is bounded and the expansion costs are minimized. In the latter problem, the goal is to minimize the total criticality in the expansion subject to budget constraints. We study the effect of the criticality constraints in these problems by simulations on example networks. We also compare the criticality constraints with the more standard $N-1$ constraints both theoretically and by simulations on the aforementioned example networks. For CME we present a greedy heuristic, which is both fast and gives very good results compared to solving the MILP formulation with Gurobi.

The remainder of this chapter is structured as follows. In the following section we define basic terms and models that we use throughout this section. In Section 4.3 we reformulate the criticality criterion by Witthaut et al. [Wit+16] and relate it to the $N-1$ criterion in a theoretical analysis. We formulate the criticality criterion as linear constraints, which are then included in a model for Transmission Network Expansion Planning resulting in models for CC-TNEP and CME. We develop a greedy heuristic for CME in Section 4.4. In Section 4.5 we evaluate and compare the presented models and algorithms on example networks. We further compare the criticality criterion and the $N-1$ criterion on these example networks. We finally conclude with a summary of the results and give an outlook on potential future work on this topic in Section 4.6.

4.2 Preliminaries

Recall the definition of flow networks and expansions in Chapter 2. In particular, in a network on a graph $G = (V, E)$ each vertex $v \in V$ has a specified demand interval $D(v)$. In this section, we extend these definitions to networks with multiple time periods. What this means is that there is a set T of timestamps, and instead of a single demand function D we have one demand function D_τ per timestamp $\tau \in T$. Thus, for every $\tau \in T$ we get a network $\mathcal{N}_\tau$ on the same graph G. In this chapter all such networks have fixed demands. Hence, we specify functions $d_\tau \colon V \to \mathbb{R}$ instead of the functions D_τ, which assign demand intervals to the vertices.

Formally, let $G = (V, E)$ be an orientation of an undirected graph. Each edge $e \in E$ has a *capacity* $\mathrm{cap}(e) \in \mathbb{R}_{>0}$ and a susceptance $b(e) \in \mathbb{R}_{>0}$. Additionally, there is a set of timestamps T. For each timestamp $\tau \in T$ and vertex $v \in V$, the demand $d_\tau(v)$ at v is fixed. We call the input tuple $(G, T, \{d_\tau\}_{\tau \in T}, \mathrm{cap}, b)$ a *multi-period network*.

Given such a multi-period network $\mathcal{N}$ with timestamps T, we define the *network at time* τ by $\mathcal{N}_\tau = (G, d_\tau, \mathrm{cap}, b)$. A multi-period network $\mathcal{N}$ *admits an electrical flow* if there is an electrical flow in every network $\mathcal{N}_\tau$ for $\tau \in T$.

Expansions for multi-period networks and their costs can be defined as for single-period networks. This requires the edge set E to be partitioned into a set of *existing edges* E_{ex} and a set of *candidate edges* E_{cand}. An expansion of a network $\mathcal{N}$ is any subnetwork of $\mathcal{N}$ with edgeset E_H, where $E_{\mathrm{ex}} \subseteq E_H \subseteq E$.

In this chapter, we call the tuple consisting of a multi-period network $\mathcal{N}$, a partioning of the edge set into existing and candidate edges, and a function $c \colon E_{\mathrm{cand}} \to \mathbb{R}_{\geq 0}$ an *instance*. The cost for building a candidate edge $e \in E_{\mathrm{cand}}$ is given by $c(e)$.

Finding the cheapest expansion that admits an electrical flow of a (single-period) network is the Electrical Flow Expansion problem; see Section 2.2. Clearly, we can generalize this problem to expansions of multi-period networks.

Definition 4.1 (Multi-Period Electrical Flow Expansion (MP-EFE)). Given an instance on a multi-period network $\mathcal{N}$ with timestamps T, find an expansion $\mathcal{H}$ admitting an electrical flow at all times $\tau \in T$ and that has minimum cost among all such expansions.

As usual, we denote the cost of $\mathcal{H}$ by $\mathrm{cost}(\mathcal{H})$. This problem lies at the heart of most Transmission Network Expansion Planning problems. Since MP-EFE is an extension of Electrical Flow Expansion, Proposition 3.4 directly implies that it is $\mathcal{NP}$-hard to determine whether a multi-period network has an expansion.

In order to formulate this problem as an MILP, we introduce the binary variables $z(e)$ for all $e \in E_{\mathrm{cand}}$. We interpret $z(e) = 1$ as "the candidate e is selected in the expansion", and $z(e) = 0$ as "it is not selected in the expansion". Further, we have the continuous variables $f_\tau(e)$ for all $\tau \in T$ and all $e \in E$ representing the electrical flows

and $\theta_\tau(v)$ for all $\tau \in T$ and all $v \in V$ representing the angles . The conservation of flow (see Equation (2.2)) is described by the linear equations for all $\tau \in T$ and $v \in V$ by

$$\sum_{uv \in E} f_\tau(uv) - \sum_{vw \in E} f_\tau(vw) = d_\tau(v). \tag{4.1}$$

For the existing edges the capacity constraints and Kirchhoff's voltage law are described as in Equations (2.1) and (2.6). That is, we have for all $\tau \in T$ and $vw \in E_{\text{ex}}$

$$|f_\tau(vw)| \leq \text{cap}(vw), \tag{4.2}$$
$$f_\tau(vw) = b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)). \tag{4.3}$$

The flow on a candidate edge vw has to obey the same conditions as on existing edges if the edge is selected in the expansion ($z(vw) = 1$), and it must be 0 if vw is not selected ($z(vw) = 0$). We therefore have for all $\tau \in T$ and all $vw \in E_{\text{cand}}$

$$|f_\tau(vw)| \leq \text{cap}(vw) \cdot z(vw), \tag{4.4}$$
$$f_\tau(vw) = z(vw) \cdot b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)). \tag{4.5}$$

Equation (4.4) ensures that the flow on the edges does not exceed the capacity. Equation (4.5) requires Kichhoff's Voltage Law to be satisfied if $z(vw) = 1$, and places no restriction on the angles if $z(vw) = 0$. The latter equation is non-linear, but it can be linearized using big-M-constraints.

$$|f_\tau(vw) - b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w))| \leq M_{vw} \cdot (1 - z(vw)). \tag{4.6}$$

The minimal values for M_{vw} can be determined by computing shortest paths in G equipped with a suitable metric [BPG01]. Now, Equations (4.1) to (4.4) and (4.6) represent the basic constraints for the MP-EFE problem. The objective is to minimize

$$\sum_{vw \in E_{\text{cand}}} z(vw) \cdot c(vw). \tag{4.7}$$

This basic version can be extended in various ways, e.g., by additionally minimizing the operation costs (e.g., [CMT07, ARK11, ATCM12]), or by considering that lines may be added over a longer time horizon [KSK10, ARK11, ATCM12]. However, such extensions are out of scope in this chapter. We focus on the effect of including a criticality measure into expansion planning problems.

4.3 The Criticality Criterion

Witthaut et al. [Wit+16] propose and evaluate simple classifiers for determining whether the failure of a line in a power grid causes the grid to desynchronize. In this chapter we consider a classifier based on maximal (graph-theoretical) flows in the residual networks. In the following sections we reformulate this classifier in graph-theoretical terms (Section 4.3.1) and compare it to the $N-1$ criterion (Section 4.3.2). We formulate it as a set of linear constraints (Section 4.3.3), which we then include in an MILP formulation for extensions of MP-EFE (Section 4.3.4).

4.3.1 Formulation of the Criticality Criterion

Given an expansion $\mathcal{H}$ with the graph $H = (V, E_H)$ let f_τ be the electrical flow in $\mathcal{H}$ at time $\tau \in T$ and let $vw \in E_H$ be an edge. The *residual network* $\mathcal{N}^{\text{res}}(f_\tau, vw)$ consists of a directed graph with vertex set $V^{\text{res}} := V$ and edge set E^{res}. For each edge $xy \in E_H \setminus \{vw\}$ the residual network contains the edges xy and yx. The *residual capacity* $\text{cap}^{\text{res}}(xy)$ of an edge $xy \in E^{\text{res}}$ is $\text{cap}(xy) - f_\tau(xy)$ if $xy \in E_H$ and $\text{cap}(yx) + f_\tau(yx)$ if $yx \in E_H$. We further require that the flow on each directed residual edge is at least 0. That is, flow on the residual edges is only allowed in the direction of the edges. We denote the total flow value of the maximum (graph-theoretical) flow in $\mathcal{N}^{\text{res}}(f_\tau, vw)$ from v to w by $F^{\text{res}}_\tau(v, w)$. Similarly, $F^{\text{res}}_\tau(w, v)$ is the maximum total flow value in $\mathcal{N}^{\text{res}}(f_\tau, vw)$ from w to v.

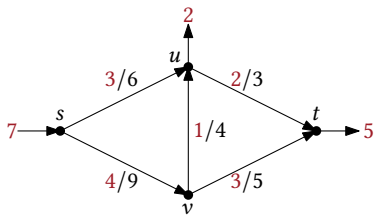
Figure 4.1 shows an example electrical flow and three residual networks. In Figure 4.1 (a) the electrical flow in the network is shown. The demands at the vertices are indicated by arrows pointing into the vertices (sources with negative demands) and away from them (sinks with positive demands). Figures 4.1 (b), (c), and (d) show the residual networks for the edges su, sv, and vu, respectively. Note that even though only one direction of each edge is shown, the residual networks actually contain edges in both directions; showing both directions and their capacities would clutter the illustration.

Definition 4.2. The *criticality* of an edge $vw \in E_H$ at time $\tau \in T$ is defined by

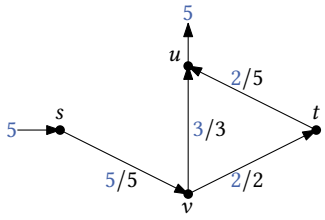
$$\text{crit}_\tau(vw) = \begin{cases} \max\{0, f_\tau(vw) - h \cdot F^{\text{res}}_\tau(v, w)\}, & f_\tau(vw) \geq 0, \\ \max\{0, -f_\tau(vw) - h \cdot F^{\text{res}}_\tau(w, v)\}, & f_\tau(vw) < 0, \end{cases}$$

where f_τ is the electrical flow in $\mathcal{H}$ at time $\tau \in T$ and $h \in \mathbb{R}_{>0}$ is a parameter.

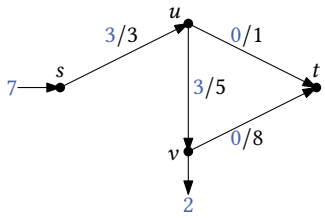
The edge vw is *critical* if $\text{crit}_\tau(vw) > 0$. If $f_\tau(vw) \geq 0$, being critical is equivalent to $f_\tau(vw)/F^{\text{res}}_\tau(v, w) > h$. In that sense, the parameter h represents a threshold when we classify an edge as critical.



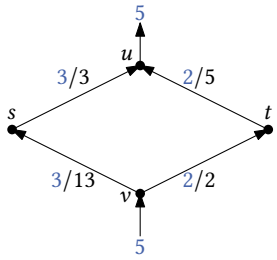
(a) The electrical flow.



(b) $\mathcal{N}^{\text{res}}(f, su)$, $\frac{3}{5} \leq h$.



(c) $\mathcal{N}^{\text{res}}(f, sv)$, $\frac{4}{3} > h$.



(d) $\mathcal{N}^{\text{res}}(f, vu)$, $\frac{1}{5} \leq h$.

Figure 4.1: An example graph and three residual networks. The edges are marked by $f(e)/\text{cap}(e)$. The susceptance of all edges is 1. Vertices with negative/positive demands are marked by arrows into/out of the vertices. In the residual networks only the direction of the edges carrying flow is shown, and we assume $h = 1$. Electrical flows are written in red, and graph-theoretical flows in blue. (a) The network with its electrical flow. (b) The residual network $\mathcal{N}^{\text{res}}(f, su)$ admits a maximum flow of $F^{\text{res}}(f, s, u) = 5 \geq 3 = h \cdot f(su)$. Thus, su is not critical. (c) $F^{\text{res}}(f, s, v) = 3 < 4 = h \cdot f(sv)$. The criticality of sv is 1. (d) $F^{\text{res}}(f, v, u) = 5 \geq 1$. The edge vu is not critical.

In the example in Figure 4.1, we assume $h = 1$ for simplicity. For the edge su (Figure 4.1 (b)) we see that $f_\tau(su) = 3 \leq 5 = h \cdot F^{\text{res}}(f, s, u)$. Hence, su is not critical. In contrast, the edge sv is critical (Figure 4.1 (c)). The maximum residual flow $F^{\text{res}}(f, s, v)$ is only 3, which is not sufficient as $f_\tau(sv) = 4$. Hence, its criticality is $\text{crit}_\tau(sv) = \max\{0, 4 - 1 \cdot 3\} = 1$.

Note that the criticality of an edge does not directly depend on the capacity of the edge but rather on the residual capacities of the other edges. Hence, criticality is different to the line congestion level, which relates the flow on an edge to the capacity of the edge [YL20].

The criticality has the same unit as the flow on the edges. In this context, the flow typically describes the power transferred via the edge. Hence, the flow and the

criticality are measured in MW. The criticality can be interpreted as the amount by which the flow on vw shall be reduced until the edge is not critical anymore. From a different point of view, $\text{crit}_\tau(vw)/h$ can be interpreted as by how much the maximum flow in the residual network needs to be increased until vw is not critical anymore. In the example of Figure 4.1 (c) this means that we would need to increase the maximum flow in the residual network $\mathcal{N}^{\text{res}}(f_\tau, s, v)$ by at least $\text{crit}_\tau(sv)/h = 1$ in order to make sv non-critical.

The *total criticality* of the expansion $\mathcal{H}$ induced by the edges E_H is defined as the sum of the criticalities over all edges and all timestamps,

$$\text{crit}(\mathcal{H}) := \sum_{\tau \in T} \sum_{vw \in E_H} \text{crit}_\tau(vw).$$

The equation above directly gives a way to compute the total criticality of an expansion $\mathcal{H}$. Note that for each timestamp the electrical flow in $\mathcal{H}$ only needs to be computed once. In total, this yields a running time of $\mathcal{O}(|T| \cdot (T_{\text{EF}} + m \cdot T_{\text{MF}}))$, for an expansion with m edges, where T_{EF} and T_{MF} are the running times for computing an electrical flow and a (graph-theoretical) maximum flow. With the bounds from Sections 2.1 and 2.1.2 we obtain the following lemma.

Lemma 4.3. *The total criticality of an expansion $\mathcal{H}$ with m edges can be computed in $\mathcal{O}(|T| \cdot m^{5/2})$ time.*

We observe that the value of the maximum flow in the residual network is at least 0. If $f_\tau(vw) < 0$, we therefore have $\max\{0, f_\tau(vw) - h \cdot F_\tau^{\text{res}}(v, w)\} = 0$. Similarly, we obtain $\max\{0, -f_\tau(vw) - h \cdot F_\tau^{\text{res}}(wv)\} = 0$ if $f_\tau(vw) \geq 0$. Hence, we can equivalently compute the criticality of an edge vw as follows.

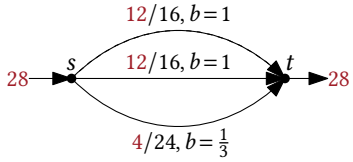
Lemma 4.4.

$$\text{crit}_\tau(vw) = \max \left\{ \begin{array}{l} 0, \\ f_\tau(vw) - h \cdot F_\tau^{\text{res}}(v, w), \\ -f_\tau(vw) - h \cdot F_\tau^{\text{res}}(w, v) \end{array} \right\}.$$

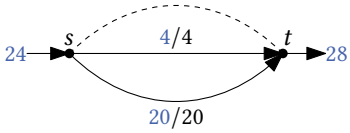
This equation lends itself more easily to be formulated as a linear program than the original formulation. We therefore base our linear constraints on this formulation; see Section 4.3.3.

4.3.2 Relation to the $N-1$ Criterion

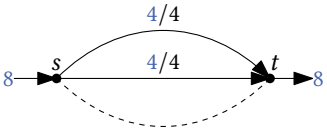
An expansion $\mathcal{H}$ satisfies the $N-1$ criterion under edge failures if and only if removing one edge from $\mathcal{H}$ still yields a network that admits an electrical flow [ZA05]. As the criticality criterion considers edge failures (and not vertex or other equipment



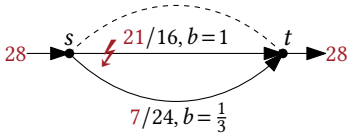
(a) The network $\mathcal{N}$ with electrical flow f.



(b) $\mathcal{N}^{\text{res}}(f, e_1)$, $12/24 \leq h$.



(c) $\mathcal{N}^{\text{res}}(f, e_3)$, $4/8 \leq h$.



(d) No electrical flow in $\mathcal{N} - e_1$.

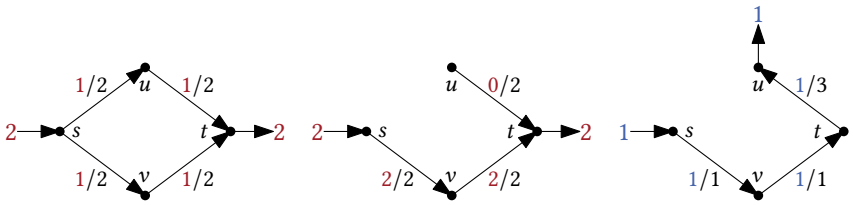
Figure 4.2: An example network without critical edges for $h \geq 0.5$, but removing any edge prohibits an electrical flow. (a) The electrical flow f in the network. The vertex s acts as a source with demand $d(s) = -28$, and t acts as a sink with demand $d(t) = 28$. The edges are marked by $f(e)/\text{cap}(e), b(e)$. (b) The maximum residual flow in $\mathcal{N}^{\text{res}}(f, e_1)$ has value $24 \geq f(e_1)/h = 12/h$. (c) The maximum residual flow in $\mathcal{N}^{\text{res}}(f, e_3)$ has value $8 \geq f(e_3)/h = 4/h$. (d) If e_1 fails there is no electrical flow in the resulting network because the edge e_2 is overloaded.

failures), we restrict ourselves to the $N-1$ criterion under edge failures. In the remainder of this chapter, we simply call this the $N-1$ criterion without explicitly mentioning edge failures.

It is similar to the criticality criterion by Witthaut et al. [Wit+16] in the sense that both consider the failure of one edge. Both criteria aim to establish whether such an edge failure causes the network to fail. However, in the $N-1$ criterion only the static behavior is considered. The network after one edge failure must still admit a (static) electrical flow. The dynamics of the failing edge are ignored. In contrast, the criticality criterion tries to capture whether one failing edge causes the network to desynchronize.

In general, if a network satisfies one of the two criteria, it does not need to satisfy the other. Figures 4.2 and 4.3 show networks that satisfy one criterion but not the other. However, we shall see in Section 4.5.4 that the two criteria are related empirically.

The network in Figure 4.2 has no critical edge for $h \geq 0.5$, which in particular includes the value of 0.614 determined by Witthaut et al. [Wit+16]. If either of the edges, say e_i, is removed, the maximum residual flow is $2f(e_i)$. The maximum



(a) The network $\mathcal{N}$. **(b)** $\mathcal{N} - su$. **(c)** $\mathcal{N}^{\text{res}}(f, su)$.

Figure 4.3: An example network, in which all edges are critical, but which still admits an electrical flow even if one edge is removed. (a) The electrical flow f in the network. The vertex s acts as a source with a demand of $d(s) = -2$ and t acts as a sink with $d(t) = 2$. The edges are marked by $f(e)/\text{cap}(e)$. All edges have susceptance 1. (b) The electrical flow after the edge su fails. (c) The maximum flow in the residual network $\mathcal{N}^{\text{res}}(f, s, u)$ has value 1. For $h < 1$, the edge su is critical.

residual flows in $\mathcal{N}^{\text{res}}(f, e_1)$ and $\mathcal{N}^{\text{res}}(f, e_3)$ are shown in Figures 4.2(b) and 4.2(c); the case of removing e_2 is symmetric to the case of removing e_1. But if any of the edges fails, the remaining network does not admit an electrical flow. For example, if e_1 fails (see Figure 4.2(d)), Kirchhoff's voltage law would require a flow on e_2 that is larger than the capacity of e_2.

Conversely, the network in Figure 4.3 satisfies the $N-1$ criterion because if either of the four edges fails, the other path is sufficient to transport the required 2 units of power from s to t; see Figure 4.3(b). However, for $h < 1$ all edges are critical. For example, if the edge su is removed, the maximum residual flow from s to u is 1, but the required residual flow is $f(su)/h = 1/h > 1$; see Figure 4.3(c).

However, in the case that we choose $h \geq 1$, then any network that satisfies the $N-1$ criterion also satisfies the criticality criterion. To see this consider the electrical flows f and f' before and after an edge vw is removed. As the network satisfies the $N-1$ criterion both flows exist. We may assume without loss of generality that $f(vw) \geq 0$. Then, the difference f'' defined by $f'' = f' - f$ is a flow in the residual network $\mathcal{N}^{\text{res}}(f, vw)$ with value $f(vw)$. Hence,

$$f(vw) \leq F^{\text{res}}(v, w) \leq h \cdot F^{\text{res}}(v, w),$$

and thus vw is not critical.

Note, however, that Witthaut et al. [Wit+16] empirically determined a value of $h = 0.614$ in their case study. While the optimal choice of h may depend on the graph topology, their value is far less than 1. Hence, we expect that one should choose $h < 1$ in realistic networks. This means that we are in the regime where

neither the criticality criterion implies the $N-1$ criterion nor vice versa. Hence, it may make sense to consider both criteria together.

4.3.3 The Criticality Criterion as Linear Constraints

In this section we present how the criticality can be incorporated as part of a (mixed-integer) linear program. In Section 4.3.4 we describe how to actually incorporate these constraints in MP-EFE models. The formulation of the criticality constraints is not limited to expansion planning problems. It may also be incorporated in other transmission network design problems, for example Optimal Transmission Switching [FOF08].

The criticality of an edge vw at time $\tau \in T$ depends on the maximum flow in the residual flow network $\mathcal{N}^{\text{res}}(f_\tau, vw)$, where f_τ is the electrical flow in the original network at time τ. In $\mathcal{N}^{\text{res}}(f_\tau, vw)$ the vertices v and w act as unbounded sources and sinks. That is, the consumption at v and w must lie in $(-\infty, 0]$ and $[0, \infty)$, respectively. At all other vertices the amount of flow entering the vertex equals the flow leaving it. Note that in our model we do not explicitly set one of the vertices v and w as a source and the other as a sink. Their roles are implicitly determined when optimizing the model. This is different to the original formulation of the criticality condition. In this formulation v acts as a source if there is positive flow from v to w (i.e., $f_\tau(vw) > 0$) and w acts as a source if there is positive flow from w to v (i.e., $f_\tau(vw) < 0$).

Suppose $f_\tau \colon E_H \to \mathbb{R}$ is an electrical flow in the expansion $\mathcal{H}$. We model the criticality of an edge $xy \in E_H$ as follows. We have one continuous variable $f^{\text{res}}_{vw,\tau}(xy)$ for each edge $vw \in E_H$, which models the residual flow on xy. As for the electrical flow, we interpret positive values of $f^{\text{res}}_{vw,\tau}(xy)$ as flow from x to y and negative values as $-f^{\text{res}}_{vw,\tau}(xy)$ units flowing from y to x. There further is one continuous variable $c_\tau(vw)$ representing the criticality of the edge vw.[1]

For the ease of presentation, we consider the edge vw and the time τ as fixed, and drop the subscripts vw and τ. That is, we write $f^{\text{res}}(xy)$ and $c(xy)$ instead of $f^{\text{res}}_{vw,\tau}(xy)$ and $c_\tau(xy)$. In our models, all constraints below are repeated for all edges vw and all time stamps τ. We first model the capacity constraints for the residual flow.

$$f^{\text{res}}(vw) = 0, \tag{4.8}$$

$$f^{\text{res}}(xy) \leq \text{cap}(xy) - f(xy) \qquad \forall xy \in E_{\text{ex}}, \tag{4.9}$$

$$f^{\text{res}}(xy) \geq -\text{cap}(xy) - f(xy) \qquad \forall xy \in E_{\text{ex}}, \tag{4.10}$$

$$f^{\text{res}}(xy) \leq (\text{cap}(xy) - f(xy)) \cdot z(xy) \qquad \forall xy \in E_{\text{cand}}, \tag{4.11}$$

$$f^{\text{res}}(xy) \geq (-\text{cap}(xy) - f(xy)) \cdot z(xy) \qquad \forall xy \in E_{\text{cand}}. \tag{4.12}$$

[1] Actually, the constraints only ensure that $c_\tau(vw)$ is an upper bound for the criticality of vw. As we minimize over $c_\tau(vw)$, we may think of it as representing the criticality of vw.

Equation (4.8) ensures that there is no residual flow on vw since we consider the residual flow network where vw is removed. Equations (4.9) and (4.10) restrict the flow on existing edges to the residual capacities. The residual capacity constraints for the candidate edges are modeled by Equations (4.11) and (4.12). This includes requiring a flow of 0 on candidate edges that are not included in the resulting expansion ($z(xy) = 0$). The last two equations are non-linear, but they can be linearized in the following way.

$$f^{\text{res}}(xy) \leq 2\text{cap}(xy) \cdot z(xy) \qquad \forall xy \in E_{\text{cand}}, \tag{4.13}$$
$$f^{\text{res}}(xy) \geq -2\text{cap}(xy) \cdot z(xy) \qquad \forall xy \in E_{\text{cand}}, \tag{4.14}$$
$$f^{\text{res}}(xy) \leq \text{cap}(xy) - f(xy) \qquad \forall xy \in E_{\text{cand}}, \tag{4.15}$$
$$f^{\text{res}}(xy) \geq -\text{cap}(xy) - f(xy) \qquad \forall xy \in E_{\text{cand}}. \tag{4.16}$$

The first two inequalities ensure that if the edge $xy \in E_{\text{cand}}$ is not in the final solution ($z(xy) = 0$), then $f^{\text{res}}(xy) = 0$. If $z(xy) = 1$, the last two inequalities ensure that the residual capacity of xy is not exceeded. Note that in this case the first two equations do not restrict $f^{\text{res}}(xy)$ any further since $|f(xy)| \leq \text{cap}(xy)$.

As stated above we ensure that the flow is conserved at all vertices except at the endpoints v and w of the edge vw. For those two vertices we impose no restriction on their consumption.

$$\sum_{xu \in E} f^{\text{res}}(xu) - \sum_{uy \in E} f^{\text{res}}(uy) = 0 \qquad \forall u \in V \setminus \{v, w\}. \tag{4.17}$$

Equations (4.9), (4.10) and (4.13) to (4.17) model a flow from v to w (or vice versa) in the residual network $\mathcal{N}^{\text{res}}(f, vw)$. So far, however, we do not require any minimum flow between v and w. In particular, setting all variables $f^{\text{res}}(xy)$ to 0 satisfies all constraints.

We base the constraints for the criticality of the edge vw, represented by $c(vw)$, on the criticality formulation in Lemma 4.4.

$$c(vw) \geq 0, \tag{4.18}$$
$$c(vw) \geq f(vw) - h \cdot \left(\sum_{xw \in E} f^{\text{res}}(xw) - \sum_{wy \in E} f^{\text{res}}(wy) \right), \tag{4.19}$$
$$c(vw) \geq -f(vw) - h \cdot \left(\sum_{xv \in E} f^{\text{res}}(xv) - \sum_{vy \in E} f^{\text{res}}(vy) \right). \tag{4.20}$$

Here, h is the threshold used to classify an edge as critical (see Definition 4.2). These constraints ensure that $c(vw)$ is at least the criticality of the edge vw in the resulting expansion.

4.3.4 Criticality in Transmission Network Expansion Planning

Having seen how to formulate the criticality criterion as a set of linear constraints, we show how to include them in a MP-EFE model. There are two direct ways to modify the MP-EFE problem. The first way is to require that the resulting expansion does not exceed a given maximum total criticality Crit_{max}. Of course, we also require the expansion to be feasible. Recall that an expansion $\mathcal{H}$ of an electrical flow network is feasible if and only if there is an electrical flow in $\mathcal{H}$.

Definition 4.5 (Criticality-Constrained Transmission Network Expansion Planning (CC-TNEP)). Given an instance with a network $\mathcal{N}$ and a maximum total criticality $\text{Crit}_{\text{max}} \in \mathbb{R}_{\geq 0}$, find a feasible expansion $\mathcal{H}$ with $\text{crit}(\mathcal{H}) \leq \text{Crit}_{\text{max}}$ of minimum costs.

Alternatively, we can directly consider the total criticality of the expansion as our objective.

Definition 4.6 (Criticality Minimal Expansion (CME)). Given an instance with a network $\mathcal{N}$ and a budget $\text{Cost}_{\text{max}} \in \mathbb{R}_{\geq 0}$, find a feasible expansion $\mathcal{H}$ with $c(\mathcal{H}) \leq \text{Cost}_{\text{max}}$ that admits a feasible electrical flow and has minimum total criticality.

Adding Equations (4.8) to (4.20) to the basic MP-EFE problem and requiring that the resulting total criticality is at most $\text{Crit}_{\text{max}} \in \mathbb{R}_{\geq 0}$, i.e.,

$$\sum_{\tau \in T} \sum_{vw \in E} c_\tau(vw) \leq \text{Crit}_{\text{max}}, \tag{4.21}$$

we obtain an MILP-formulation of the CC-TNEP problem; see Appendix A.1 for a presentation of all constraints together. Recall that in any solutions the values of the variables $c_\tau(vw)$ are just an upper bound for $\text{crit}_\tau(vw)$. But with Equation (4.21) this implies that the total criticality of the resulting expansion is at most Crit_{max}.

Similarly, we can model CME as an MILP, taking the basic constraints of MP-EFE as well as the criticality constraints (Equations (4.8) to (4.20)). Different to CC-TNEP, we require the total cost of the expansion to be bounded by Cost_{max}, i.e.,

$$\sum_{vw \in E_{\text{cand}}} z(vw) \cdot c(vw) \leq \text{Cost}_{\text{max}}, \tag{4.22}$$

and we minimize the total criticality, which is

$$\sum_{\tau \in T} \sum_{vw \in E} c_\tau(vw). \tag{4.23}$$

A full presentation of the model is given in Appendix A.2. Note that as we minimize over the sum of the variables $c_\tau(vw)$, we have $c_\tau(vw) = \mathrm{crit}_\tau(vw)$ in any optimal solution. Hence, the objective value of an optimal solution can directly be interpreted as the criticality of the resulting expansion.

4.4 A Greedy Heuristic for Criticality Minimal Expansion

In addition to the MILP formulation for CME, we develop a simple greedy heuristic. We start with the network $\mathcal{N}_0$ on the graph $G_0 = (V, E_{\mathrm{ex}})$, which contains all existing edges but no candidate edges. The initial remaining budget r_0 is $\mathrm{Cost}_{\max}$. At each step i, we have an expansion $\mathcal{N}_i$ induced by an edge set E'_i and budget r_i. We then determine which candidate reduces the total criticality the most if it is added to $\mathcal{N}_i$. To this end, we consider each candidate edge $e \in E_{\mathrm{cand}}$ that has not been selected in the expansion $\mathcal{N}_i$, i.e., $e \notin E'_i$. If $c(e) > r_i$, the candidate edge is too expensive, and we ignore it. Otherwise, we compute the total criticality of the expansion induced by $E'_i \cup \{e\}$.

Afterwards, if there is a candidate that reduces the total criticality, we choose the edge e for which the total criticality of the expansion induced by $E'_i \cup \{e\}$ is minimal. We then set $E'_{i+1} := E'_i \cup \{e\}$ and $r_{i+1} := r_i - c(e)$. If all candidates are either too expensive or do not reduce the total criticality, we stop and return $\mathcal{N}_i$ as the resulting expansion. In particular, if we have reached an expansion without critical edges, we stop since no expansion further reduces the total criticality.

Note that it may happen that an expansion is not feasible since it does not admit an electrical flow. If this happens during the check whether added an edge is worthwhile, we simply ignore this expansion. However, if the initial network $\mathcal{N}_0$ already does not admit an electrical flow, we proceed differently. We define the *total capacity violation* $\mathrm{viol}(E')$ of an expansion $\mathcal{H}'$ with edge set E' by

$$\mathrm{viol}(E') = \sum_{\tau \in T} \sum_{e \in E'} \max\{0, |f'_\tau(e)| - \mathrm{cap}(e)\}, \tag{4.24}$$

where f'_τ is an electrical flow in $\mathcal{H}'$ at time τ except that it may violate some edge capacity constraints. Recall that Kirchhoff's voltage law (Equation (2.6)) uniquely determines f'_τ [Bol98]. Thus, $\mathrm{viol}(E')$ is well-defined.

We now proceed similar to the main part of the algorithm. But instead of minimizing the total criticality, we greedily minimize the total capacity violation. If we reach a point where the resulting expansion admits an electrical flow, we switch to greedily minimizing the total criticality.

Table 4.1: The properties of the networks in the evaluation. The candidate edges are parallel to the existing edges.

Country	$\lvert V\rvert$	$\lvert E_{\text{ex}}\rvert$	Country	$\lvert V\rvert$	$\lvert E_{\text{ex}}\rvert$
AT	23	29	IE	8	12
BE	27	32	NL	33	40
BG	12	17	NO	41	65
CH	15	23	PT	16	22
CZ	21	35	RO	17	27
DK	11	11	SE	46	72
HR	6	6	SI	4	4
HU	13	21	SK	9	13

Lemma 4.7. *The greedy algorithm runs in* $\mathcal{O}(|E_{\text{cand}}|^2 \cdot |E|^{5/2})$ *time.*

Proof. The edge set of expansion grows by one candidate per iteration. Hence, there are at most $|E_{\text{cand}}|$ iterations. In each iteration we compute the total criticalities of at most $|E_{\text{cand}}|$ expansions. To compute the total criticality of single expansion we perform $|E|$ maximum-flow computations, each taking $\mathcal{O}(|E|^{3/2})$ time (see Section 2.1.1). □

4.5 Evaluation

Based on the theoretical analysis of the models we formulate hypotheses that guide our evaluation. These hypotheses are then verified or falsified empirically on 16 sample networks that are extracted from the data available in PyPSA [BHS18]. Each network is a clustered version of the transmission grid of one European country. The networks have between 4 and 46 vertices, and between 4 and 72 edges; see Table 4.1 for details. There is a candidate edge parallel to each existing edge, i.e., the total number of edges in the graph is twice the number of existing edges. The data for the maximum generation and load at each vertex are available in hourly resolution over the course of one year. We restricted our evaluation to four days. To alleviate the impact of seasonal and weekly variation, we chose one Tuesday in winter (22 January 2013) and the following Sunday (27 January) and the same in summer (16 July, 21 July).

In the data, however, only the maximum generation is available and not the actual generation. But different generation distributions induce different electrical flows

and thus different criticality values. To exclude the impact of different generation distributions, we fixed the generations based on a merit order principle. We assigned the required power to the cheapest generators. While an optimal power flow [FR16] may be more desirable from an optimization point of view, it has the disadvantage that it depends on the network topology. Since the topology changes when adding edges, one would have to re-calculate the optimal power flow. This makes the optimization more complex. Moreover, using the merit order is realistic in the sense that it is used to decide which generators are active, e.g., in the European Union [Gom+19].

The models and the heuristic presented above are able to deal with multiple timestamps. We were interested how the number of timestamps considered together affects the solution. Therefore, we split the 96 timestamps in total in groups of k timestamps each. For this we chose $k \in \{1, 2, 3, 4, 6, 12, 24\}$. That is, for each network we had $96/k$ groups. For the criticality threshold parameter h, we use $h = 0.614$, which is the value Witthaut et al. [Wit+16] determined as optimal in their case study.

We implemented our algorithms in C++17, compiled with GCC 8.2.1, and used Gurobi 9.0.0 [Gur22] to solve the MILPs. The algorithms were executed on a server with 64-bit architecture, four 12-core AMD CPUs running at 2.1 GHz, 256 GB of RAM under openSUSE Leap 15.1. We ran tests on 40 instances in parallel, but each algorithm was only allowed to use a single thread. The latter was done to ensure a fair comparison between Gurobi and our heuristic, which is unable to utilize multiple threads.

We gave the Gurobi one hour time to solve the models. In cases where this was not sufficient to prove optimality or that the instance is infeasible, we report the best solution found within this time frame.

4.5.1 MP-EFE vs. CC-TNEP

In a first step we assess the effect of including the criticality constraints in the MP-EFE model. To this end we compare solving the plain MP-EFE model to solving the CC-TNEP model. For CC-TNEP, we choose $\mathrm{Crit}_{\max} = 0$, which means that we require the resulting expansions to have not critical edges. We expect Gurobi to obtain the optimal results faster for the MP-EFE model than for the CC-TNEP model as the CC-TNEP model is more complex.

Hypothesis 4.8. Optimal solutions for MP-EFE can be obtained faster than optimal solutions for CC-TNEP.

The number of constraints grows quadratically in the number of edges for CC-TNEP but only linearly for MP-EFE. Hence, we expect the ratio between the solution times for CC-TNEP and MP-EFE to grow with increasing network sizes.

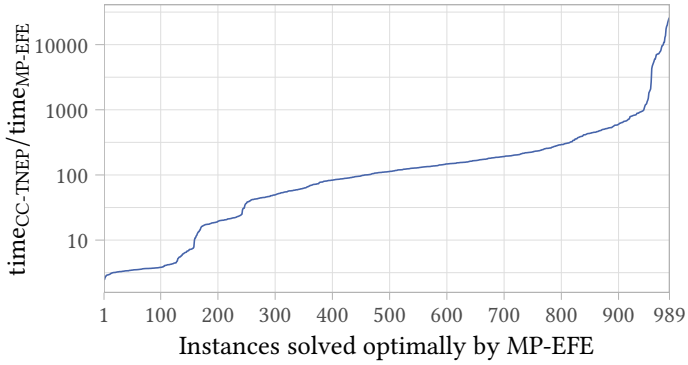


Figure 4.4: The ratio of the solution times for CC-TNEP to the solution times for MP-EFE on the 989 instances with one timestamp where both MP-EFE and CC-TNEP found solutions. The instances are sorted by increasing ratio.

Hypothesis 4.9. The ratio between the solution times for CC-TNEP and MP-EFE increases with increasing number of edges.

To verify or falsify these hypotheses, we compare solving MP-EFE and CC-TNEP on the instances with one timestamp. There are 96 timestamps for each of the 16 countries. Hence, we have 1536 instances in total. Out of these, there are 336 instances where no expansion admits an electrical flow. That is, MP-EFE (and consequently CC-TNEP) has no feasible solution. For one other instance Gurobi was not able to find any solution in one hour. Ignoring these instances, we have 1199 instances left. For all these instances, Gurobi was able to find optimal solutions for MP-EFE. Out of these 1199 instances, 209 instances do not admit an expansion without any critical edges. Additionally, there is one instance for which Gurobi was unable to find any feasible solution of CC-TNEP in one hour. This leaves us with 989 instances for which Gurobi found optimal solutions of both MP-EFE and CC-TNEP.

In the following analysis, we focus on how long it takes until the optimal solution is found by Gurobi and not until Gurobi can actually prove optimality. We therefore use the *solution time*, which we define as the time when Gurobi found the optimal solution.

Figure 4.4 shows the ratio of solution times for CC-TNEP and for MP-EFE. The instances are sorted by increasing ratio. The minimum solution time ratio is about 2.5, which means that Gurobi takes more than twice as long on all instances. The median and maximum ratios are 111.5, and 25 940. Plots for more than one timestamp have a similar shape; see Appendix A.3. These findings confirm Hypothesis 4.8.

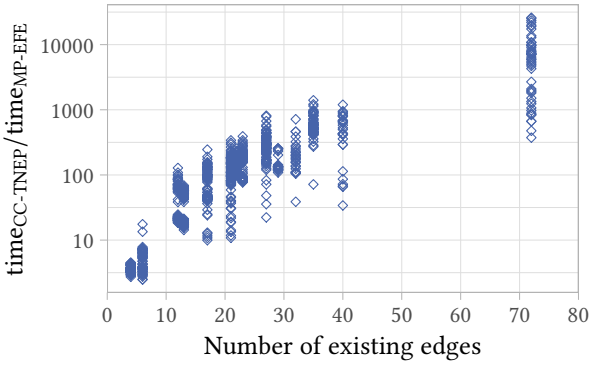


Figure 4.5: The ratio of the solution times for CC-TNEP to MP-EFE plotted against the number of existing edges. Each mark corresponds to one instance. Only instances with one timestamp where both MP-EFE and CC-TNEP found solutions are shown.

In absolute terms, however, both models could be solved quite fast. The solution times for MP-EFE are all below one second (with a maximum of 670 ms). For CC-TNEP slightly more than half of the instances (528, 53.4 %) could be solved within one second, and only 49 instances (5.0 %) needed more than one minute.

When we plot the solution time ratio in comparison to the number of edges (Figure 4.5), we can clearly see a trend that this ratio becomes larger the more edges there are. This observation confirms Hypothesis 4.9.

4.5.2 CC-TNEP vs. CME

CME has a budget as an additional parameter. For our tests, we set the budget to certain fractions (5 %, 10 %, 15 %, 20 %, 25 %, 50 %) of the total costs of all candidate edges.

The mixed-integer linear programs for CC-TNEP and CME are similar as the objective for one problem has a hard bound in the other problem and vice versa. We therefore expect the running times to be similar.

Hypothesis 4.10. On the same instances the running times for CME and CC-TNEP are the same.

To evaluate this hypothesis, we analyze those instances for which Gurobi found a feasible solution within one hour for both problems. These are between 958 instances for a 5 %-budget and 984 instances for both 10 %- and 15 %-budgets. We plot the ratio of the solution times of Gurobi for CME vs. CC-TNEP in Figure 4.6. Each line shows

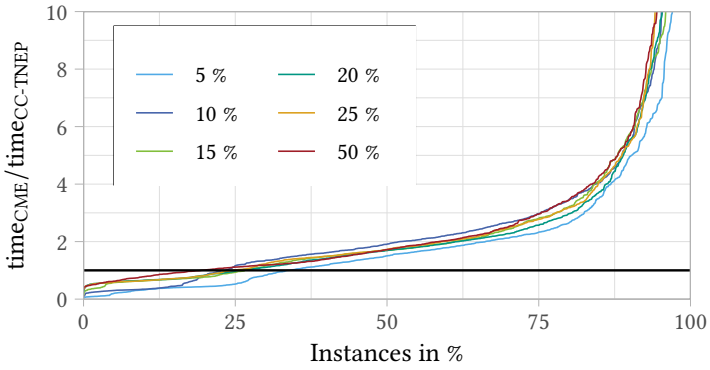


Figure 4.6: The ratio of the solution times for CME vs. CC-TNEP on the instances with one timestamp. For each budget, the instances are sorted by increasing ratio.

the solution time ratio for one of the budgets; the instances are sorted by these ratios individually for each line. We can see that except for the 5 %-budget, CME can be solved faster than CC-TNEP on slightly less than 25 % of the instances. But on most instances solving CC-TNEP is faster. On 40 % of the instances the ratio exceeds 2, compared to only 8.0 % of the instances with a ratio below 0.5. We therefore conclude that solving CME is on average slower than solving CC-TNEP and therefore reject Hypothesis 4.10

4.5.3 Evalution of the Greedy Heuristic for CME

We now compare solving the CME model with Gurobi to running the greedy heuristic presented in Section 4.4. When Gurobi is given enough time, we are guaranteed to find the optimal solution if the instance is feasible. The greedy algorithm, however, does not have this guarantee. It may even fail to find any feasible solution if the existing network does not admit an electrical flow. But as the greedy heuristic tries to deal with this case, we nevertheless expect it to usually find feasible solutions.

Hypothesis 4.11. There are instances where Gurobi finds a solution but the greedy algorithm does not.

Moreover, we expect the greedy algorithm to find good solutions in much less time than Gurobi.

Hypothesis 4.12. The greedy algorithm finds optimal solutions on the majority of instances.

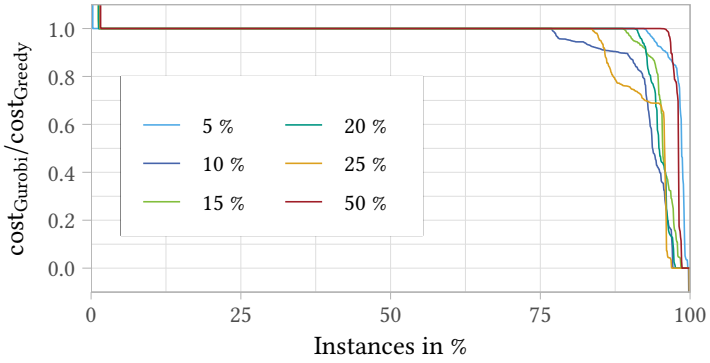


Figure 4.7: The ratio of the total criticalities of the expansions computed by Gurobi for CME vs. the greedy algorithm on the instances with one timestamp. Instances where neither method found a solution are omitted. For each budget, the instances are sorted by decreasing ratio.

Hypothesis 4.13. The greedy algorithm is faster than Gurobi.

The running time of the greedy algorithm depends only linearly on the number of timestamps. While the model size also grows linearly with the number of timestamps, the decision which candidates to add depends on all timestamps, and having more timestamps allows for more complex interactions. Thus, we expect the running time of Gurobi to increase super-linearly with the number of timestamps. As a consequence, we expect the time advantage of the greedy heuristic compared to Gurobi to increase if the number of timestamps increases.

Hypothesis 4.14. The time advantage of the greedy algorithm compared to Gurobi increases with the number of timestamps.

Similarly, the running time of the greedy algorithm only indirectly depends on the budget. If it finds a solution with total criticality 0 early, it stops independent on the budget available. However, if more budget is available the space of feasible solutions becomes larger. This may impede Gurobi in finding good solutions.

Hypothesis 4.15. The time advantage of the greedy algorithm compared to Gurobi increases with the budget.

Among the 9216 instances with one timestamp (over all six budget choices) there are 2221 infeasible instances and 9 instances where neither Gurobi nor the greedy algorithm found any solution. We use the remaining 6986 instances in the following analysis. Gurobi solved 6914 instances (99.0 %) optimally within one hour. There are

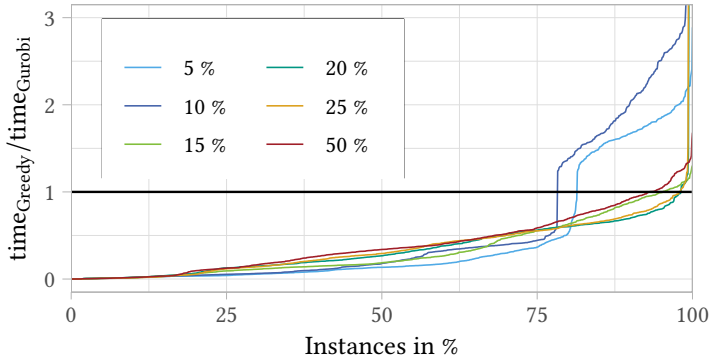


Figure 4.8: The ratio of solution times for the greedy algorithm and Gurobi solving CME on instances with one timestamp. Only the instances where both solution methods found the same result are plotted. For each budget, the instances are sorted by increasing ratio.

7 instances (0.1 %) where Gurobi found a solution (the optimal solution in all cases) but the greedy algorithm was unable to find any feasible solution. Conversely, there are 43 instances (0.6 %), where the greedy algorithm found a feasible solution but Gurobi did not. On most instances both Gurobi and the greedy algorithm found a solution. In fact, on 86.6 % of the instances they computed expansions with the same criticality. This observation is also visible in Figure 4.7, where the ratio of the total criticalities for the expansions computed by the greedy algorithm and Gurobi are plotted. Each line corresponds to one budget. For each budget choice, the instances are sorted by decreasing ratios. Values below 1 mean that the expansion computed by Gurobi has a smaller total criticality than the one by the greedy algorithm. It is clearly visible that for most instances the ratio is 1. This observation confirms Hypotheses 4.11 and 4.12. That is, the greedy algorithm is competitive to the MILP solver Gurobi in terms of resulting total criticality.

To assess the running time, we plot the ratio of the solution times of the greedy algorithm to the solution times of Gurobi. As being fast but providing much worse results is not useful, we consider only those instances where Gurobi and the greedy algorithm provide expansions with the same total criticality; see Figure 4.8. As before, each line in the plot corresponds to one budget choice. We observe that the greedy algorithm is faster on 91.0 % of the instances over all budget choices, but for small budgets this portion is notably smaller: 81.4 % for a 5 %-budget and 78.2 % for a 10 %-budget. On slightly more than half of the instances (51.0 %) the greedy algorithm is

faster by a factor of at least 4; and on 28.4 % of the instances it is faster by a factor of at least 10. These results confirm Hypothesis 4.13. Regarding the dependence on the budget, however, our results only support Hypothesis 4.15 insofar as there are more instances for smaller budgets where the greedy algorithm is slower. For well over 75 % of the instances, the budget does not make any significant difference.

For a larger number of timestamps, the time advantage of the greedy algorithm increases; see Appendix A.4. For three timestamps, it is faster by a factor of 10 on already more than half of the instances (59.4 %). This portion raises to 66.7 % for instances with six timestamps. Hence, we confirm Hypothesis 4.14.

4.5.4 $N-1$ Criterion and Criticality

In Section 4.3.2 we observe that in general the $N-1$ criterion and the criticality criterion do not imply each other. However, both criteria require redundancy in the resulting expansion. Hence, we expect expansions without critical edges to often satisfy the $N-1$ criterion as well. In an expansion $\mathcal{H}$, we call an edge e *vital* if $\mathcal{H}-e$ does not admit an electrical flow. Hence, an expansion satisfies the $N-1$ criterion if and only if it has no vital edges. The *fraction of vital edges* of an expansion $\mathcal{H}$ is the number of vital edges divided by the total number of edges in $\mathcal{H}$.

Hypothesis 4.16. An optimal solution to CC-TNEP with $\mathrm{Crit}_{\max} = 0$ tends to satisfy the $N-1$ criterion, i.e., its fraction of vital edges is close to 0.

Compared to the cost-minimal expansions (i.e., optimal solutions of MP-EFE), we expect cost-minimal expansions with total criticality 0 to have fewer vital edges.

Hypothesis 4.17. If $\mathcal{H}_{\text{MP-EFE}}$ and $\mathcal{H}_{\text{CC-TNEP}}$ are optimal solutions for MP-EFE and CC-TNEP, then the fraction of vital edges in $\mathcal{H}_{\text{CC-TNEP}}$ is at most the fraction of vital edges in $\mathcal{H}_{\text{MP-EFE}}$.

To verify or falsify these hypotheses, we consider the expansions that result from CC-TNEP and compare them to cost-minimal expansions, which result from solving MP-EFE. Expansions for instances with multiple timestamps at once tend to include more candidate edges than the expansions for the individual timestamps. Hence, they should be more likely to have few vital edges. In this sense, the instances with only one timestamp are the hardest instances. We therefore focus on those. As in Section 4.5.1, we only consider the 989 instances solved by Gurobi for both MP-EFE and CC-TNEP.

We find that there is no instance for which the fraction of vital edges is larger in the expansion computed by CC-TNEP than in the cost-minimal expansion. This supports Hypothesis 4.17. In Figure 4.9 each line corresponds to the expansions computed by solving either MP-EFE or CC-TNEP. For each line the instances are sorted by

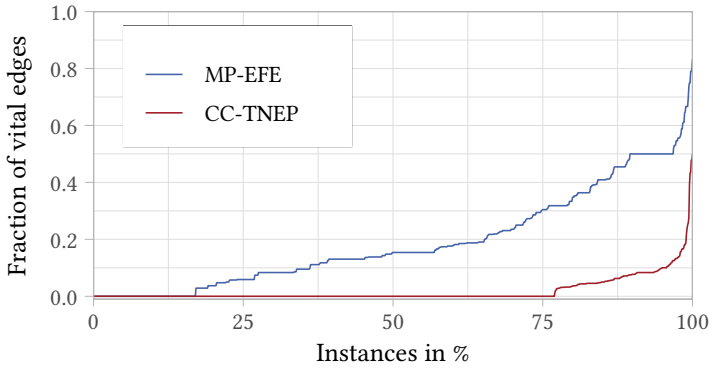


Figure 4.9: The fraction of vital edges in optimal expansions as computed by solving MP-EFE and CC-TNEP. Only instances with one timestamp are shown.

their fraction of vital edges. Note, however, that the values shown at the same x-coordinate do not necessarily correspond to the same instance as the instances are sorted per curve. We observe that for 76.9 % of the instances, the expansion computed by CC-TNEP has no vital edge. That is, the expansion satisfies the $N-1$ criterion. For MP-EFE, this holds only for 17.1 % of the instances. Hence, we conclude that requiring no critical edges tends to result in expansions that satisfy the $N-1$ criterion. We thus confirm Hypothesis 4.16.

4.6 Conclusion

In this chapter we present how to extend any (mixed-integer) linear program formulation of any transmission network design problem by the criticality criterion introduced by Witthaut et al. [Wit+16]. To this end we formulate the criticality criterion as a set of linear constraints. These may then be used as a building block when formulating transmission network optimization problems. To introduce those constraints only variables (or constants) representing the electrical flow on each edge and at each timestamp are needed.

As an example we analyzed the effects of adding the criticality criterion to MP-EFE, which is a basic version of the Transmission Network Expansion Planning problem. We formulated two problems: CC-TNEP, where the total criticality is bounded by a hard constraint, and CME, where the criticality is minimized. We further present a greedy heuristic for CME, which is both fast and produces optimal

solutions in more than 75 % of the instances. We further observed that minimizing the criticality subject to a budget constraint (CME) seems to be harder than minimizing the cost subject to a maximum criticality (CC-TNEP).

While models including the criticality criterion take longer to solve, the resulting networks are much more reliable. In particular, they tend to satisfy the $N-1$ criterion as evidenced by our simulations. Hence, one may also consider using the criticality criterion instead of (or in addition to) the $N-1$ criterion in transmission network design problems.

It would be interesting to include the criticality constraints in more comprehensive variants of Transmission Network Expansion Planning; for example, by including the operation costs in the optimization criterion or by considering expansions over a longer time scale. Moreover, the criticality constraints can be incorporated in other transmission network optimization problems, e.g., Optimal Power Flow [FR16] or Optimal Transmission Switching [FOF08]. One may analyze the applicability of the criticality criterion in online settings such as transmission grid operation. One could also investigate larger networks or analyze the relation to other reliability criteria such as the $N-k$ criterion [MSA15].

5 Algorithmic Approaches for Microgrid Cable Layouts

Microgrids play an important role in the electrification of rural areas. Designing a microgrid comprises multiple parts including finding suitable sites for the generation units, sizing the components of the microgrid, and determining the layout of the cables that connect the components. In this chapter we focus on the latter part, which we formalize as the Microgrid Cable Layout problem. We assume that the locations and the sizes of the generators and consumers are given as points in the plane. The goal is to find a cost-minimal cable layout that is able to handle the demands of the consumers. An important difference to the network design problems in the previous chapters is that we are not given a set of candidate edges. Instead, the edges, which represent cables, may be placed anywhere in the plane. In fact, we are even allowed to introduce vertices that are neither located at a generator nor at a consumer. Moreover, each edge is assigned a cable type from a given set of cable types. We prove that Microgrid Cable Layout is a strongly $\mathcal{NP}$-hard, non-linear optimization problem.

Furthermore, we present a hybrid genetic algorithm for the Microgrid Cable Layout problem. The topology (represented by a graph) is optimized by a genetic algorithm, and a heuristic assigns the cable types to the edges of the topology. An evaluation on a set of benchmark instances indicates that our algorithm is able to find better solutions within a short amount of time than heuristics for related network design problems. Furthermore, we evaluate the performance of the algorithm in a case study on a real-world microgrid in the Democratic Republic of the Congo.

This chapter is based on joint work with Max Göttlicher [GW22].

5.1 Introduction

In 2015 the UN have presented an agenda for sustainable development [Uni15b], which formulates 17 goals to reach until 2030. SDG 7 is to "ensure access to affordable, reliable, sustainable and modern energy for all" [Uni15b, p. 14]. Part of this goal is to provide everyone with access to electricity. According to a case study in rural Kenya [KJKM09] access to electricity significantly improves the productivity per worker by up to 200 %. Providing access to electricity is particularly challenging in rural areas in the global south. According to the 2021 report on the sustainable development goals [Uni21] there were still 471 million people in rural areas of sub-Saharan Africa without access to electricity in 2019.

In particular in these rural areas microgrids can play a crucial role in providing access to electricity. They do not require possibly long and costly transmission lines to the main power grid to be built. Instead, they can be installed anywhere and work completely autonomously provided that there is an energy source available. Traditionally, diesel generators were used to power microgrids, but also more sustainable sources like small hydropower plants, photovoltaic systems, or wind turbines can be used. They do not require expensive fuel, but are less reliable since they depend on external factors, e.g., wind speed or solar irradiance. Not having to build costly connections to the national transmission grids often makes microgrids an economical choice as studies for Western Australia [Fle+17] and for sample of 50 countries (including both developing and developed ones) [HU16] show.

Designing a microgrid is a complex task (see e.g., the guide by Sumanik-Leary et al. [Sum+14]), and one needs to answer many questions such as: Where should the microgrid be built? Which places shall be connected to the microgrid? How much power is needed? Which generators shall be placed where? How does the cable layout look like? In this work we focus on the last question, which deals with routing the cables efficiently. The cable layout of microgrids in remote areas is often acyclic [RFRW22]. We formalize this problem as the Microgrid Cable Layout problem. In this problem we assume the locations and sizes of the generators and consumers to be fixed by a previous step of the microgrid design process. The goal is to find a cost-minimal cable layout that connects these locations subject to certain electrical constraints. There are multiple cable types to choose from, and we may introduce additional distribution nodes, which may be placed not only at the location of a generator or consumer in the input but also at any other point in the plane. Microgrid Cable Layout further includes costs for poles, which need to be placed to support the cables. We give a formal definition of this problem in Section 5.3. Note that this problem should be thought of as a part of the full microgrid design problem. While solving the full problem, multiple instances of the Microgrid Cable Layout problem may need to be solved.

Contribution and Outline. We present a hybrid genetic algorithm for the Microgrid Cable Layout problem, with which we aim to achieve the following two main goals.

1. The algorithm should find good solutions reasonably quickly such that it can be included as a part of a software that considers the full microgrid design problem.
2. The algorithm should be flexible, i.e., it should be easy to add further constraints that need to be considered. It is infeasible to gather a set of constraints beforehand that suffices for all possible use cases.

In Section 5.2 we give an overview over the literature on microgrid planning, focusing in particular on works that consider the cable layout. We then formally define the Microgrid Cable Layout problem in Section 5.3 and prove that it is $\mathcal{NP}$-hard in Section 5.4. We describe our hybrid genetic algorithm for this problem in Section 5.5. Afterwards, we evaluate it on a set of synthetic benchmark instances (Section 5.6). This evaluation includes a case study on a real microgrid in the Democratic Republic of the Kongo. Finally, we conclude this chapter with a short summary and possible further research directions in Section 5.7.

5.2 Related Work

Determining good designs for microgrids encompasses a lot of different steps, which range from determining the size of the components over finding a good topology to computing a suitable strategy to control the operation of the microgrid. For an overview on this large range of topics we refer to two surveys by Al-Ismail [Al-21] and by Gamarra and Guerrero [GG15]. In the following we focus on works that are related to the cable layout design aspect of microgrid design.

Lambert and Hittle [LH00] tackle the problem of designing a good low- and medium voltage grid layout for the electrification of a rural village. They consider the design of the low-voltage layout and of the medium-voltage layout as two separate layers of their optimization algorithm. They use a single cable in each layer and limit resistive losses by constraining the maximum distance between demand points and transformers. For the low-voltage layout they propose to use an approach based on simulated annealing. Their medium-voltage layout is always a minimum spanning tree between the transformers, which connect the two voltage levels. The number and placement of the transformers is part of their optimization problem. However, these transformers may only be placed at demand nodes.

Kahveci et al. [KOPS16] present a heuristic for finding cable layouts for microgrids. Their heuristic includes three steps: computing a minimum spanning tree, adding

Steiner points in triangles if they reduce the costs, and finding clusters that may form independent islands. They insert Steiner points to reduce the total network length but do not take into account different cable types or electric constraints.

Corigliano et al. [CCEM20] present an approach to identify good electrification strategies in rural ares. Within this approach they apply an algorithm to compute good grid layouts considering the terrain. They first discretize the problem by defining a grid graph with weights that represent the difficulty of the terrain. Then, they compute a minimum spanning tree between the populated vertices (the metric they use is not stated explicitly however), and finally, they replace the edges in the tree by shortest paths in the grid. Similar approaches were used in case studies in Nigeria [BCB19] and in the Philippines [BC19]. Another case study for a rural and an urban microgrid in India [PRW18] uses Homer [HOM22] to optimize the sizing of the components but does not consider the geographic layout.

In contrast to the works above, Nolan et al. [NSRF17] consider multiple cable types. They present a genetic algorithm using Prim-predecessor encoding [LG06]. Unlike our algorithm, their algorithm does not consider adding additional points to the network. That is, their resulting network topology is always a spanning tree on the input points.

Before being able to compute cable layouts, one needs to decide which points should be connected. Put more abstractly, given a set of consumers, decide which subsets of points shall be part of the same microgrid. With clustering algorithms this decision can be made before designing the individual microgrids [Che+17]. This decision may, however, also be combined with determining the topologies of the microgrids. Rosenberg et al. [RFRW22] present two evolutionary algorithms for the combined problem.

A complementary version of the microgrid layout problem we consider in this work is studied by Vallem and Mitra [VM05] and Vallem, Mitra, and Patra [VMP06]. They consider the problem where to place distributed generation units given a grid topology. They solve this problem with a simulated annealing approach. Note that this approach is not directly comparable to the one we use since it solves a different problem. We assume the locations of the generation units to be fixed in order to include cases where, e.g., the generators already exist or where there is only one sensible place for a small hydropower plant.

The Microgrid Cable Layout problem is closely related to other geometric layout optimization problems such as the Euclidean Minimum Steiner Tree problem.[2] A survey on the history of this problem is written by Brazil et al. [BGTZ14]. Genetic algorithms have been used both for this problem [Bar03] and the Capacitated Minimum Spanning Tree problem in graphs [JJM04], another related network design problem. For the Euclidean Minimum Spanning Tree problem with restricted vertex degree a hybrid genetic algorithm exists [SS20]. Here, there is some input value

[2] The (Euclidean) Minimum Steiner Tree problem is often called the Steiner Minimum Tree problem and abbreviated by (E)SMT to distinguish it from the Minimum Spanning Tree problem (MST). We stick to the name *Minimum Steiner Tree* and prevent ambiguities by always writing the full problem name.

$d \in \mathbb{N}$ such that all vertices of the spanning tree have a degree of at most d. Recently, a genetic algorithm for Euclidean Minimum Steiner Tree with both soft and hard obstacles was presented [RFRW21]. Hard obstacles are regions that the resulting tree may not intersect. In contrast, the regions representing soft obstacles may be used, but they increase the cost of edges overlapping the obstacles.

5.3 The Microgrid Cable Layout Problem

As input we are given a set P of points in the plane, which represent the positions of the available generators and the consumers. Each point $p \in P$ has a *maximum power generation* $g(p) \in \mathbb{R}_{\geq 0}$ and a *maximum power demand* $d(p) \in \mathbb{R}_{\geq 0}$. Note that this setting is different from the other chapters. It makes the notation easier to explicitly distinguish between generation and demands here, unlike in the other chapters where the generation is represented by negative demands. We may assume that for each point p one of $g(p)$ and $d(p)$ is non-zero since we can ignore all points where both are zero. A point $p \in P$ is a *generator* if $g(p) > 0$ and a *consumer* if $d(p) > 0$. Moreover, we are given a set C of cable types. Each cable type $c \in C$ has a cost per meter $c_{\text{line}}(c)$, a resistivity $\rho(c)$ measured in $\Omega\,\text{m}^{-1}$, and a thermal capacity $\text{cap}(c)$ measured in kW, which represents the maximum power that can be transferred via a cable of type c.

The structure of an electricity grid can be described by a graph. This graph $(V, \overline{E})$ is inherently undirected. However, the flows on the graph are inherently directed Therefore, we represent $(V, \overline{E})$ by the directed graph (V, E), where $E = \{vw, wv \mid \{v, w\} \in E\}$. Note again that this differs from the other chapters, where only one of vw and wv is present in a network. A *topology* is then such a directed graph $G = (V, E)$ with $P \subseteq V \subseteq \mathbb{R}^2$, i.e., the vertices are a set of points in the plane that contains P. Borrowing from the nomenclature for Steiner trees, we call the points in $V \setminus P$ *Steiner points*. A topology G is a *tree* if the underlying undirected graph $\overline{G}$ is a tree, i.e., it is connected and has no cycles. Due to our focus on cost minimization in this chapter, we only consider topologies that are trees. Including cycles would increase the costs (but mitigate the effects of line failures). The length $\ell(e)$ of an edge $e \in E$ is the euclidean distance between its endpoints. A *cable assignment* is a function $a\colon E \to C$ such that $a(vw) = a(wv)$ for all $vw \in E$. That is, every edge and its reverse are assigned the same cable type. In particular, each edge is assigned exactly one cable type. Note however that placing multiple cables in parallel can be modeled by including cable types in C that represent placing multiple cables of (actual) cable types in parallel.

Typically, the installed power generation capacity, which is equal to the sum of the maximum power generations at all points, is less than the total maximum demand. That is, based on the generation capacity, we cannot fulfill all demands if all points try

to consume their maximum demands at the same time. But we want to prevent the grid infrastructure and especially the cables from being the limiting factor. Hence, we want to design the grid such that it allows for all possible distributions of generation and consumption within the bounds given by the maximum generations and demands. This does not only include the cable power rating but also losses of power and voltage within the network [Sum+14, Lou18].

The goal to handle all (and in particular infinitely many) potential demand distributions makes this setting different to the other chapters, where we consider only finitely many demand distributions. The constraints we present below are based on a book by Louie [Lou18, Sec. 3.2.3].

To determine the most extreme situations that may occur, we compute for each directed edge $vw \in E$ the maximum power $p(vw)$ that may be transmitted via vw from v to w. Recall that for $p \in P$ the values $g(p)$ and $d(p)$ give bounds for the *maximum* generation and *maximum* demand at p, respectively. The actual generation and consumption may (and usually will) differ from these values. These two functions can naturally be extended to V by setting their values to 0 outside of P. As G is a tree, removing an edge vw and its reverse wv disconnects G into two components G^v_{vw} and G^w_{vw}, where the former contains v and the latter contains w. The maximum flow $p_G(vw)$ on the edge from v to w is limited by both the total generation in G^v_{vw} and the total demand in G^w_{vw}. We have

$$p_G(vw) = \min\left\{\sum_{x \in V(G^v_{vw})} g(x), \sum_{x \in V(G^w_{vw})} d(x)\right\}. \tag{5.1}$$

Note that in general $p_G(vw)$ and $p_G(wv)$ may differ. We can compute these values for all edges in $\mathcal{O}(|V|)$ time by performing a single depth-first search of G; see Section 5.5.2 for details. In the following we omit the subscript G from p_G and other values, leaving the dependency on G implicit, provided that the context makes clear which graph is meant.

Since we want the cables to not limit the amount of power transmitted, we must ensure that their capacity is large enough in any cable assignment a, i.e., for every edge $vw \in E$ we require

$$\max\{p(vw), p(wv)\} \leq \operatorname{cap}(a(vw)). \tag{5.2}$$

We further want the line losses and the maximum voltage drop along any generator-consumer-path to be small. For a given topology $G = (V, E)$ and a cable assignment a we bound these values as follows. The maximum voltage drop $u_{\text{drop}}(vw)$ along a single edge $vw \in E$ with length $\ell(vw)$ in a three-phase transmission system is given

by $r(vw) \cdot i(vw)$, where

$$r(vw) = \ell(vw)\rho(a(vw)),$$
$$i(vw) = p(vw)/(\sqrt{3}U\cos\varphi).$$

Here, U is the grid voltage, $\cos\varphi$ is the power factor. The value $r(vw)$ describes the resistance of the edge as defined by its length and the resistivity of the assigned cable type. The value $i(vw)$ is the maximum current on vw from v to w. We can then bound the voltage drop along a directed path Q in G by

$$\begin{aligned} U_{\text{drop}}(Q) &= \sum_{vw \in E(Q)} u_{\text{drop}}(vw) \\ &= \sum_{vw \in E(Q)} \ell(vw)\rho(a(vw)) \cdot \frac{p(vw)}{\sqrt{3}U\cos\varphi}, \end{aligned}$$

where $E(Q)$ are the edges of Q. The maximum power loss $p_{\text{loss}}(vw)$ on a single edge $vw \in E$ is given by

$$p_{\text{loss}}(vw) = 3r(vw) \cdot i_{\max}(vw)^2,$$

where $i_{\max}(vw) = \max\{i(vw), i(wv)\}$ is the maximum current along the edge vw in either direction. The sum of all these values is an upper bound for the total power loss P_{loss}.

$$\begin{aligned} P_{\text{loss}} &= \sum_{vw \in E} p_{\text{loss}}(vw) \\ &= \sum_{vw \in E} \ell(vw)\rho(a(vw)) \cdot \left(\frac{\max\{p(vw), p(wv)\}}{U\cos\varphi}\right)^2. \end{aligned}$$

Our goal is find a topology G with a cable assignment a of minimum costs such that the maximum voltage drop along any generator-consumer-path is at most $\alpha_{\text{drop}} \cdot U$, and the total line losses P_{loss} are at most a factor of α_{loss} of the total maximum generation and consumption. More formally, we want for each directed path Q from a generator to a consumer that

$$U_{\text{drop}}(Q) \leq \alpha_{\text{drop}} \cdot U \tag{5.3}$$

and

$$P_{\text{loss}} \leq \alpha_{\text{loss}} \cdot \min\left\{\sum_{p \in P} g(p), \sum_{p \in P} d(p)\right\} \tag{5.4}$$

We call any cable assignment that satisfies Equations (5.2) to (5.4) a *feasible* cable assignment.

A topology $G = (V, E)$ with cable assignment a incurs certain costs. In this chapter we consider the costs for the cables, the poles the cables are placed on, and the equipment needed to connect the cables together. We assume that these include both the costs of the materials and the installation. Some costs like the costs of the generators depend neither on the topology nor on the cable assignment and can be ignored for the purpose of determining a good topology and cable assignment. Note that the algorithms described in the following section are designed to be flexible such that additional costs may be introduced easily if the need arises.

To compute that cable costs, recall that each cable $c \in C$ has a cost per meter $c_{\text{line}}(c)$. Hence, we have

$$\text{cost}_{\text{cables}}(G, a) = \sum_{e \in E} \ell(e) \cdot c_{\text{line}}(a(e)).$$

The cables are supported by poles, which may be at most $\ell_{\max}$ (measured in m) apart. There is one pole placed at each vertex. If an edge is longer than $\ell_{\max}$, we additionally need to place poles in the interior of the edge. To ensure that the distance between two such consecutive poles is at most $\ell_{\max}$, we place on each edge $e \in E$

$$\left\lceil \frac{\ell(e)}{\ell_{\max}} \right\rceil - 1$$

poles, not counting the poles at the endpoints of the edges. We assume a fixed cost of c_{pole} per pole. In total, the cost of the poles is given by

$$\text{cost}_{\text{poles}}(G) = c_{\text{pole}} \cdot \left(|V| + \sum_{e \in E} \left\lceil \frac{\ell(e)}{\ell_{\max}} \right\rceil - |E| \right).$$

Further we consider the equipment at each vertex that is necessary to connect the incident lines. The cost of this equipment at $v \in V$ is determined by a function c_{equip}, which we assume to be given as an input. This function depends on the degree $\deg_G(v)$ of v and the maximum power on any edge incident to v, which is defined as

$$\hat{p}_G(v) = \max\{p_G(vw), p_G(wv) \mid w \in N_G(v)\},$$

where $N_G(v)$ denotes the neighbors of v in G. We then have

$$\text{cost}_{\text{vertex}}(G) = \sum_{v \in V} c_{\text{equip}}(\deg_G(v), \hat{p}_G(v)).$$

In our experiments we assume that c_{equip} depends linearly on its parameters, i.e., we have

$$c_{\text{equip}}(x, y) = \alpha + \beta \cdot x + \gamma \cdot y,$$

for some values $\alpha, \beta, \gamma \in \mathbb{R}_{\geq 0}$. In total, the costs are

$$\text{cost}(G, a) = \text{cost}_{\text{cables}}(G, a) + \text{cost}_{\text{poles}}(G) + \text{cost}_{\text{vertex}}(G).$$

The Microgrid Cable Layout problem can be summarized as follows. We are given a set of points $P \subseteq \mathbb{R}^2$ with generation function $g\colon P \to \mathbb{R}_{\geq 0}$ and demand function $d\colon P \to \mathbb{R}_{\geq 0}$, a set of cables C with their properties (cost per unit c_{line}, resistivity ρ, and capacity cap), and the properties of the desired grid (voltage U, power factor $\cos\varphi$, maximum voltage drop factor α_{drop}, maximum power loss factor α_{loss}, maximum distance between poles ℓ_{max}, costs of poles c_{pole}, and costs of other equipment c_{equip}). We then want to find a topology $G = (V, E)$ that is a tree and a cable assignment $a\colon E \to C$ such that they satisfy Equations (5.2) to (5.4) and minimize $\text{cost}(G, a)$.

5.4 Hardness of the Microgrid Cable Layout Problem

The Microgrid Cable Layout problem generalizes the strongly $\mathcal{NP}$-hard Euclidean Minimum Steiner Tree problem [BGTZ14, GGJ77]. The input of the Euclidean Minimum Steiner Tree problem consists of a set of points $P \subseteq \mathbb{R}^2$ in the Euclidean plane. The goal is to connect these points by line segments (possibly ending at additional points called *Steiner points*) such that the total length of the segments is minimum. In the terms defined above we aim to find a topology such that the sum of all edge lengths is minimum.

Theorem 5.1. *It is strongly $\mathcal{NP}$-hard to decide given an instance of the Microgrid Cable Layout problem and some $x \in \mathbb{Q}_{\geq 0}$ whether the instance has a solution of cost at most x.*

Proof. We reduce the Euclidean Minimum Steiner Tree problem to the Microgrid Cable Layout problem as follows. Let $P \subseteq \mathbb{R}^2$ be the points of the Euclidean Minimum Steiner Tree instance. We use them as the points in the Microgrid Cable Layout instance we build. Let $p \in P$ be an arbitrary point, which we use as the single generator, and all other points are consumers. That is, we set $g(p) = 1$, $d(p) = 0$, and for all $p' \in P \setminus \{p\}$ we set $g(p') = 0$ and $d(p') = 1$. We have only one cable type c with $\text{cap}(c) = 1$, $\rho(c) = 1$, and $c_{\text{line}}(c) = 1$. As the total maximum generation is 1, the cable capacity is sufficient for every edge in every topology. To ignore the voltage drop and power loss requirements, we set $\alpha_{\text{drop}} = \alpha_{\text{loss}} = 1$. Hence, the resistivity of the cable does not actually matter. Moreover, we have no costs for the poles ($c_{\text{pole}} = 0$) and do not require any poles in interior of the edges by setting

$\ell_{\max}$ to some sufficiently large value. In total, for each topology $G = (V, E)$ there is a unique cable assignment a, and we have

$$\text{cost}(G, a) = \sum_{e \in E} \ell(e),$$

which is precisely the objective function of the Euclidean Minimum Steiner Tree problem. Hence, the Euclidean Minimum Steiner Tree instance and the Microgrid Cable Layout instance we constructed are equivalent. This transformation is possible in linear time. Thus, the strong $\mathcal{NP}$-hardness of Euclidean Minimum Steiner Tree implies that Microgrid Cable Layout is strongly $\mathcal{NP}$-hard as well. □

In fact, even finding a cost-minimal cable assignment given a fixed topology is already $\mathcal{NP}$-complete. More precisely, we show that the following decision problem, which we call the Cable Assignment problem, is $\mathcal{NP}$-complete. Given a topology, a set of cables, the grid properties (voltage, power factor), and some $s \in \mathbb{R}_{\geq 0}$, is there a feasible cable assignment costing at most s? We prove that this problem is $\mathcal{NP}$-complete by a reduction from the $\mathcal{NP}$-complete problem Subset Sum [GJ79]. An instance of Subset Sum consists of a set $X \subseteq \mathbb{N}$ and some $k \in \mathbb{N}$. The question then is, whether there is a subset $Y \subseteq X$ such that the sum of all elements of Y is equal to k.

Theorem 5.2. *The Cable Assignment problem is $\mathcal{NP}$-complete even if the topology is a path.*

Proof. It is not hard to see that checking whether a given cable assignment meets all constraints can be done in polynomial time. Hence, the Cable Assignment problem is in $\mathcal{NP}$.

To show that it is $\mathcal{NP}$-hard as well, we reduce Subset Sum to it. Let (X, k) be an instance of Subset Sum. The topology is a path $G = (V, E)$ of $|X|$ edges, where each element $x \in X$ corresponds to one edge e_x of length x. The start vertex s of the path has a generation of $g(s) = 1$ and no demand, and the end vertex t has a demand of $d(t) = 1$ and no generation. All other vertices $v \in V \setminus \{s, t\}$ have $g(v) = d(v) = 0$. There are two cable types c_1 and c_2 with

$$\begin{aligned} c_{\text{line}}(c_1) &= 1, & \rho(c_1) &= 2, & \text{cap}(c_1) &= 1, \\ c_{\text{line}}(c_2) &= 2, & \rho(c_2) &= 1, & \text{cap}(c_2) &= 2. \end{aligned}$$

Note that c_2 is equivalent to two cables of type c_1 in parallel. Since the topology is fixed, the costs of equipment other than the cables is constant. Hence, we may assume those costs to be 0 in this proof. Note that for $vw \in E$, we have $p(vw) = p(wv) = 1$.

Hence, the capacities of both cable types are sufficient, and we can ignore them in the remainder of this proof. We choose U and φ such that

$$2 \cdot \sum_{x \in X} x - k = \frac{\sqrt{3}}{10} U^2 \cos \varphi. \tag{5.5}$$

Moreover, we set

$$\alpha_{\text{drop}} = \frac{1}{10} \quad \text{and} \quad \alpha_{\text{loss}} = \frac{\sqrt{3}}{10 \cos \varphi}.$$

We claim that there is a cable assignment of cost at most $\sum_{x \in X} x + k$ if and only if there is a solution to the Subset Sum instance (X, k).

Suppose there is $Y \subseteq X$ such that $\sum_{y \in Y} y = k$. We claim that assigning the cables by

$$a(e_x) = \begin{cases} c_1, & x \notin Y, \\ c_2, & x \in Y. \end{cases}$$

results in a feasible cable assignment. The costs of a are

$$\begin{aligned} \text{cost}(a) &= \sum_{e \in E} \ell(e) \cdot c_{\text{line}}(a(e)) \\ &= \sum_{x \in X} x \cdot c_{\text{line}}(a(e_x)) \\ &= \sum_{x \in X \setminus Y} x \cdot 1 + \sum_{y \in Y} y \cdot 2 \\ &= \sum_{x \in X} x + \sum_{y \in Y} y \\ &= \sum_{x \in X} x + k. \end{aligned}$$

Moreover, taking into account that the power along each edge is 1, we obtain the

bound for the total power loss

$$\begin{aligned}
P_{\text{loss}} &= \sum_{e \in E} \ell(e)\rho(a(e)) \cdot \frac{1}{(U\cos\varphi)^2} \\
&= \sum_{x \in X} x\rho(a(e_x)) \cdot \frac{1}{(U\cos\varphi)^2} \\
&= \left(\sum_{x \in X \setminus Y} x \cdot 2 + \sum_{y \in Y} y \cdot 1 \right) \cdot \frac{1}{(U\cos\varphi)^2} \\
&= \left(2\sum_{x \in X} x - \sum_{y \in Y} y \right) \cdot \frac{1}{(U\cos\varphi)^2} \\
&= \left(2\sum_{x \in X} x - k \right) \cdot \frac{1}{(U\cos\varphi)^2} \\
&= \frac{\sqrt{3}}{10\cos\varphi} \\
&= \alpha_{\text{loss}} \cdot 1,
\end{aligned}$$

where the second to last equality uses Equation (5.5), and the 1 in the final row is the total generation in the grid. Hence, the cable assignment a satisfies the power loss constraint. A similar chain of equations yields

$$U_{\text{drop}}(G) = \sum_{e \in E} \ell(e)\rho(a(e)) \cdot \frac{1}{\sqrt{3}U\cos\varphi} = \alpha_{\text{drop}} \cdot U.$$

Note that the whole path G is the only path we need to check for the voltage drop constraint. Hence, the assignment a is feasible and within the desired costs.

Conversely, suppose that there is a feasible assignment a of costs at most $\sum_{x \in X} x + k$. Let $Y = \{x \in X \mid a(e_x) = c_2\}$ be the set of elements whose corresponding edge has been assigned c_2. We have

$$\begin{aligned}
\sum_{y \in Y} y &= \sum_{y \in Y} 2y + \sum_{x \in X \setminus Y} x - \sum_{x \in X} x \\
&= \text{cost}(a) - \sum_{x \in X} x \\
&\leq \sum_{x \in X} x + k - \sum_{x \in X} x \\
&= k.
\end{aligned}$$

As for the converse above, we obtain

$$2\sum_{x\in X} x - \sum_{y\in Y} y = P_{\text{loss}} \cdot (U\cos\varphi)^2.$$

Applying the power loss bound, we then get

$$\begin{aligned} 2\sum_{x\in X} x - \sum_{y\in Y} y &\leq \alpha_{\text{loss}} \cdot (U\cos\varphi)^2 \\ &= \frac{\sqrt{3}}{10\cos\varphi}(U\cos\varphi)^2 \\ &= \frac{\sqrt{3}}{10}U^2\cos\varphi \\ &= 2\cdot\sum_{x\in X} x - k, \end{aligned}$$

where the final equation holds by the definition of U and φ. Rearranging this inequality yields

$$\sum_{y\in Y} y \geq k.$$

In total, we have that the sum of Y is equal to k, which shows that the cable assignment instance and the instance of Subset Sum are equivalent.

This transformation can clearly be done in polynomial time. As Subset Sum is $\mathcal{NP}$-hard [GJ79], so is Cable Assignment Problem even if the topology is a path. □

5.5 A Hybrid Genetic Algorithm for the Microgrid Cable Layout Problem

A solution to the Microgrid Cable Layout problem consists of a topology and an assignment of cables to the edges of the topology. As stated in the introduction to this chapter, we aim to achieve two goals with our algorithm: first, it should compute good solutions fast, and second, it should be able to be adapted easily to incorporate new requirements. Finding a suitable topology is related to the Euclidean Minimum Steiner Tree problem, for which there is an efficient implementation of an exact algorithm [JWWZ18]. However, this algorithm does not work with weighted edges, which would occur due to the different cable types with different costs per unit. The introduction of Steiner nodes makes the problem difficult to solve using a general purpose optimizer such as Gurobi [Gur22] as they result in a non-convex problem.

For the Euclidean Minimum Steiner Tree and various constraint spanning tree problems (e.g., the Constrained Minimum Spanning Tree) genetic algorithms have proven to work well [Bar03, JJM04, ATCM12]. Thus, we expect a genetic algorithm to be able to find good solutions of the Microgrid Cable Layout problem reasonably quickly, thus satisfying Goal 1. Moreover, genetic algorithms are composed of several operators as well as the fitness function, which can be adapted to additional requirements fairly easily. Hence, we expect a genetic algorithm to achieve Goal 2 as well.

We adapt the genetic operators for these problems to the Microgrid Cable Layout problem. We base the operators of the genetic algorithm on the genetic algorithms by Barreiros, Moharam and Morsy [Bar03, MM17]. One major difference of Microgrid Cable Layout to the constrained spanning tree problems mentioned above is that we also need to find a cable assignment. While this may also be incorporated directly in the genetic algorithm, preliminary results indicated that employing a hybrid approach, in which a genetic algorithm computes the topology (see Section 5.5.1) but not the cable assignments may work better. Within the selection phase our genetic algorithm uses one of several cable assignment algorithms (see Section 5.5.2) to evaluate the quality of the computed topologies.

5.5.1 Genetic Algorithm for the Topology

A genetic algorithm maintains a set of individuals (the *population*) and repeatedly creates new individuals by either modifying the individuals slightly (*mutation*) or by combining parts of two individuals (*crossover*). Based on a *fitness function* it then selects some old and some new individuals to form the next *generation*; see the book by Kramer [Kra17] for a more detailed introduction to genetic algorithms.

The population of our genetic algorithm consists of topologies. Each topology is represented by a singly-linked adjacency list. To assess the quality of these topologies, we compute cable assignments according to one of the approaches that we explain in Section 5.5.2. These cable assignments then allow us to compute the costs of the topologies (with these cable assignments).

Initialization

We want the topologies in our initial population to have a reasonable structure. In particular, we want them to be trees without edge crossings, since edge crossings are likely suboptimal. However, we do want a variety of such trees in order to be able to explore the solution space. To achieve both goals we compute random spanning trees on a planar triangulation of the input points. More precisely, we use the Delaunay triangulation [BCKO08, Ch. 9] as the planar triangulation, which can be computed in

expected $\mathcal{O}(n \log n)$ time, where n is the number of points [BCKO08, Thm. 9.12]. We then obtain a random spanning tree as follows. We traverse the edges in a random order (chosen uniformly at random from all permutations of the edges) and include an edge in the tree unless its inclusion would close a cycle.

Crossover

All our crossover operations work on two *parent* topologies $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. In the *separate crossover* we choose which Steiner points and edges of the two parent topologies to keep in two separate phases. In a third phase we ensure that the created topologies are connected. Let S_1 and S_2 be the Steiner points in the two parent topologies. In the first phase we select a subset S of $\max\{|S_1|, |S_2|\}$ Steiner points from $S_1 \cup S_2$ uniformly at random, which we keep. Then, $V = P \cup S$ is the set of vertices of the new topology. Not keeping all Steiner points ensures that the number of Steiner points remains bounded.

In the second phase we select a subset of the edges to insert. This phase is based on an operator for constrained balanced trees [MM17]. However, due to the first phase an edge of a parent may have endpoints that are not present in V. We therefore map each edge $vw \in E_1 \cup E_2$ to the edge $\mu(vw)$ between the point of V closest to v and the point of V closest to w and work with the mapped edges. First, we insert all of $\mu(E_1)$. Then, we select a random subset E_2' of $\mu(E_2) \setminus \mu(E_1)$ by first selecting the desired cardinality k and then k edges uniformly at random. We insert each of these edges into the child topology, breaking each cycle C that occurs by deleting random edge on C except the newly added edge.

Finally, we check in the third phase whether the resulting topology is connected. If not, we randomly add edges between the components until the topology is connected.

In the *subtree crossover* we aim to maintain (possibly optimal) local substructures. The main idea is to take one subtree of G_2 and to add it to G_1 removing all edges of G_1 in this process that would cause cycles. We select a subtree of G_2 by first choosing an edge $vw \in E_2$ uniformly at random and then picking the subtree $T = (V_T, E_T)$ to one side of vw. We initialize the child topology with $(V_1 \cup V_T, E_T)$. Then, we consider the edges of G_1 in a random order, and insert those edges that connect disconnected components. Since G_1 is connected, the resulting topology is connected as well.

Mutations

Note that neither the initialization nor the two crossover operators introduces new Steiner points. This only happens in the mutation operators, which we describe next. All mutation operators operate on a single topology $G = (V, E)$.

The *close endpoint mutation* is an adaptation of a mutation operator for diameter constrained spanning tree problems [MM17]. The operator is parameterized by a mutation probability π_e. Before the mutation we temporarily insert Steiner points in the middle of all edges of G resulting in a topology $G' = (V', E')$. Then, each edge is considered individually and mutated with probability π_e. To mutate an edge vw, we randomly select one of its endpoints x as the first endpoint of the new edge and remove vw. We then select another point $y \in V'$ as the second endpoint, where the probability that a point z is chosen is proportional to $\text{dist}(x, z)^{-2}$. However, the insertion of xy may create a cycle and disconnect the topology. As in separate crossover, we break cycles by randomly deleting an edge on the cycle. To repair a disconnected topology, we re-insert vw. Finally, we revert the introduction of many Steiner points in the beginning by contracting all Steiner points of degree at most 2; we describe at the end of this section how to do this efficiently.

To move Steiner points we apply the following two operators. Both decide for each Steiner point independently whether to move it with probability π_s. They differ in how they move the Steiner point then. The *nudge mutation* (called *compress* by Barreiros [Bar03]) moves the Steiner point to a random point on an incident edge. The *link-length minimization* mutation tries to reduce the total weighted length of the incident edges, where we set the weight as the unit costs of the assigned cable types before the modification. We employ an iterative algorithm for finding the generalized Weber point [KK62]; the number of iteration is another input parameter of this operator. Note that link-length minimization disregards the poles on the incident edges. Moreover, during the minimization we ignore the option of assigning different cable types. Both restrictions may cause the iterative algorithm to converge to a non-optimal position.

Selection

At the end of every iteration we have a set of parent topologies and a new set of child topologies, which have been created by applying the operators. To choose which topologies to keep, we need to assess their quality, which we interpret as the total cost of the topology. Since the cost depends both on the topology and a cable assignment, we first compute a cable assignment for each topology; we present different exact and heuristic methods for this in Section 5.5.2. We then use tournament selection [Kra17] with elitist reinsertion to determine the set of topologies to keep. More precisely, 95 % of the new population is obtained by repeatedly selecting a random subset of k child topologies and inserting the cheapest of these. The remaining 5 % are the 5 % cheapest parent topologies. Keeping these implies that the quality of the best topology in the population does not decrease over time.

Pruning Steiner points

The operators may cause the topologies to contain Steiner points of degree at most 2. Such Steiner points are never necessary in an optimal topology, and they can be removed by contracting an incident edge. They can be determined by a single traversal of the tree in $\mathcal{O}(|P|)$ time. Note that removing points also reduces the number of options the operators have, e.g., for deciding where to move an edge in the close endpoint mutation. However, we artificially subdivide the edges before performing such operations. Hence, pruning Steiner points in this way has only a small impact on these operators. Pruning has the benefit that it keeps the number of Steiner points bounded by a constant factor of the number of points. Otherwise, the number of Steiner points could increase with each iteration, which degrades the performance of the operators.

5.5.2 Cable Assignment

A topology is only a part of a solution. It is also necessary to assign cables to the edges of a topology. In this section we present exact and heuristic algorithms for computing cable assignments for a topology. Throughout this section $G = (V, E)$ is the tree topology for which a cable assignment shall be computed.

We first observe that the constraints for cable assignments depend on the maximum amount of power $p(vw)$ that may flow along an edge vw; see Equation (5.1). As a first step we therefore present a linear-time algorithm that computes the values of $p(vw)$ for all edges $vw \in E$; see Algorithm 5.1.

We begin the depth-first search (DFS) at an arbitrary vertex $r \in V$ (Lines 3 and 4). When the function `computeFlow()` is called for a vertex v and its parent in the search tree u, it first recursively calls `computeFlow()` on its children (Line 8). These calls return the total amount of generation and demands within the subtree rooted at the child. After the loop, g and d contain the sum of all generations and demands in the subtree rooted at v (including the generation and demand of v). These values suffice to compute the maximum flows on the edge between v and its parent u (Lines 10 and 11). Note that in the calls to the children (and recursively to their children and so on) the correct flow values of all edges in the subtree rooted at u are determined. Hence, when the algorithm finishes all desired values are computed, and we have the following result.

Lemma 5.3. *For a given topology $G = (V, E)$, the maximum flow values can be computed in $\mathcal{O}(|V|)$ time.*

Algorithm 5.1: The depth-first search to determine the maximum flows on the edges.

Data: Tree $G = (V, E)$
Result: Maximum power flow values $p\colon E \to \mathbb{R}_{\geq 0}$

```
1  g_total ← ∑_{p∈P} g(p)
2  d_total ← ∑_{p∈P} d(p)
3  r ← arbitrary point in V
4  computeFlow(r, ⊥)
5  function computeFlow(v, u):
6      (g, d) ← (g(v), d(v))
7      for w ∈ N_G(v) \ {u} do
8          (g, d) ← (g, d) + computeFlow(w, v)
9      if u ≠ ⊥ then
10         p(uv) ← min{d, g_total − g}
11         p(vu) ← min{d_total − d, g}
12     return (g, d)
```

Exact Cable Assignment

The formulation of the constraints of the cable assignment in Section 5.3 suggests formulating the cable assignment problem as an MILP. In this formulation we have one binary variable $z(e, c) \in \{0, 1\}$ per edge $e \in \overline{E}$ of the underlying undirected graph and cable type $c \in C$. All other values that occur in the formulation are constant for a given topology and can be precomputed. In particular, the power flows on the edges can be precomputed in linear time by Lemma 5.3. We ensure that exactly one cable type is selected per edge with the following constraints.

$$\sum_{c \in C} z(e, c) = 1, \qquad \forall e \in \overline{E}. \tag{5.6}$$

The resistance of an edge $e \in \overline{E}$ can then be described by the expression

$$r(e) = \sum_{c \in C} z(e, c) \cdot \ell(e)\rho(c). \tag{5.7}$$

Replacing $\ell(e)\rho(a(e))$ in Equations (5.3) and (5.4) with the expression for $r(e)$ in Equation (5.7) above, we obtain linear constraints that ensure that the voltage drops and the line losses stay within the desired range. Note that it suffices to formulate

the voltage drop constraint only for maximal paths from a generator to a consumer, i.e., they are not part of any longer path from a generator to a consumer.

To ensure that the capacity of the selected cables is sufficiently large, we use the following constraints, which adapt Equation (5.2).

$$\max\{p(vw), p(wv)\} \leq \sum_{c \in C} \text{cap}(c) \cdot z(\{v, w\}, c) \qquad \forall \{v, w\} \in \overline{E}. \tag{5.8}$$

Finally, we want to minimize the cost of the cable assignment, which is given by

$$\text{cost}(G, a) = \sum_{e \in \overline{E}} \sum_{c \in C} \ell(e) \cdot c_{\text{line}}(c) \cdot z(e, c). \tag{5.9}$$

Note that this objective function just includes the costs that depend on the cable assignment and ignores the costs that only depend on the topology. In total, we obtain an MILP with $|\overline{E}| \cdot |C|$ binary variables.

Heuristic Cable Assignment

In the genetic algorithm that computes the topology, we repeatedly compute cable assignments for topologies in order to assess their qualities. Hence, we need a fast way to compute a good cable assignment. Even though our experiments show that the MILP can typically be solved within seconds (see Section 5.6.1), this is not fast enough for our purpose. We therefore employ a heuristic, which extends the work by Kraft [Kra20] and works as follows.

As for formulating the MILP, we first determine the maximum flows on all edges ($p(vw)$ and $p(wv)$, as well as $p_{\max}(vw) = \max\{p(vw), p(wv)\}$ for all $vw \in E$). We further compute for each edge vw the length $\hat{\ell}_{\text{path}}(vw)$ of the longest path from a generator to a consumer via vw. We describe in the following how to compute these lengths in $\mathcal{O}(|V|)$ time by traversing the tree twice. This algorithm is given in Algorithm 5.2.

Any generator-consumer-path via the edge $vw \in E$ can be decomposed into a prefix from a generator s to v, the edge vw, and a suffix from w to a consumer t. In the algorithm, we therefore compute for each edge vw the values $\hat{\ell}_g(vw)$ and $\hat{\ell}_c(vw)$, which represent the maximum length of a path from a generator to v without the edge wv and the maximum length of a path from w again without the edge wv, respectively. The maximum length of a generator-consumer-path via vw is then $\hat{\ell}_g(vw) + \ell(\{v, w\}) + \hat{\ell}_c(vw)$.

In the first two traversals of the algorithm (Lines 4 and 5), the values of $\hat{\ell}_g$ and $\hat{\ell}_c$ are computed. In the first traversal (`computeLengthsBelow`) the values $\hat{\ell}_g(wv)$ and $\hat{\ell}_c(vw)$ where v is the parent of w are determined. These correspond to paths that lie below

Algorithm 5.2: The algorithm to compute the lengths of the maximum paths containing an edge.

Data: Tree $G = (V, E)$
Result: Maximum path lenghts $\hat{\ell}_{\text{path}} \colon \overline{E} \to \mathbb{R}_{\geq 0} \cup \{-\infty\}$

$r \leftarrow$ arbitrary point in V
$\hat{\ell}_g(vw) \leftarrow -\infty \; \forall vw \in E$
$\hat{\ell}_c(vw) \leftarrow -\infty \; \forall vw \in E$
`computeLengthsBelow(`$r, \perp$`)`
`computeLengthsAbove(`$r, \perp$`)`
`computeLengths()`

function `computeLengthsBelow(`w, v`)`**:**
  for $x \in N_G(w) \setminus \{v\}$ **do**
    `computeLengthsBelow(`x, w`)`
  $\hat{\ell}_g(wv) \leftarrow \max\{\hat{\ell}_g(xw) + \ell(\{x, w\}) \mid x \in N_G(w) \setminus \{v\}\}$
  $\hat{\ell}_c(vw) \leftarrow \max\{\hat{\ell}_c(wx) + \ell(\{w, x\}) \mid x \in N_G(w) \setminus \{v\}\}$
  if $g(w) > 0$ **and** $\hat{\ell}_g(wv) = -\infty$ **then**
    $\hat{\ell}_g(wv) \leftarrow 0$
  if $d(w) > 0$ **and** $\hat{\ell}_c(vw) = -\infty$ **then**
    $\hat{\ell}_c(vw) \leftarrow 0$

function `computeLengthsAbove(`w, v`)`**:**
  for $x \in N_G(w) \setminus \{v\}$ **do**
    $\hat{\ell}_g(wx) \leftarrow \max\{\hat{\ell}_g(yw) + \ell(\{y, w\}) \mid y \in N_G(w) \setminus \{x\}\}$
    $\hat{\ell}_c(xw) \leftarrow \max\{\hat{\ell}_c(wy) + \ell(\{w, y\}) \mid y \in N_G(w) \setminus \{x\}\}$
    if $g(w) > 0$ **and** $\hat{\ell}_g(wx) = -\infty$ **then**
      $\hat{\ell}_g(wx) \leftarrow 0$
    if $d(w) > 0$ **and** $\hat{\ell}_c(xw) = -\infty$ **then**
      $\hat{\ell}_c(xw) \leftarrow 0$

function `computeLengths()`**:**
  for $\{v, w\} \in \overline{E}$ **do**
    $\hat{\ell}_{\text{path}}(\{v, w\}) \leftarrow \max\{\hat{\ell}_g(vw) + \hat{\ell}_c(vw), \hat{\ell}_c(wv) + \hat{\ell}_g(wv)\} + \ell(\{v, w\})$

w in the tree. In the second traversal (`computeLengthsAbove`) the remaining values are computed. Based on these values, the desired maximum path lengths are obtained. (`computeLengths`). The second traversal and the final iteration may be combined in one traversal, but this does not change the asymptotic running time of the algorithm.

Note that a naive implementation of the maximum computations in Lines 18 and 19 requires linear time per computation, which results in quadratic running time of the algorithm in total. However, observe that at each point $v \in V$ only the two highest values in $\{\hat{\ell}_g(yw) + \ell(\{y, w\}) \mid y \in N_G(w)\}$ are relevant (and likewise for $\{\hat{\ell}_c(wy) + \ell(\{w, y\}) \mid y \in N_G(w)\}$). Hence, we can compute these values before, which allows us to compute the maxima in constant time each. This results in a linear running time for the whole algorithm.

Lemma 5.4. *For all edges $e \in E$ together the maximum lengths of generator-consumer-paths that contain an edge e can be computed in $\mathcal{O}(|V|)$ time.*

We then allocate each edge $\{v, w\}$ a portion of the allowed voltage drop that is proportional to the length of the edge relative to the length of the longest path with $\{v, w\}$. That is, we set

$$u_{\text{drop}}^{\max}(\{v, w\}) = \frac{\ell(\{v, w\})}{\hat{\ell}_{\text{path}}(\{v, w\})} \alpha_{\text{drop}} U.$$

Similarly, we allocate a portion of the allowed line losses proportional to the edge length, i.e.,

$$p_{\text{loss}}^{\max}(\{v, w\}) = \frac{\ell(\{v, ws\})}{\sum_{e \in \overline{E}} \ell(e)} \alpha_{\text{loss}} \cdot \min \left\{ \sum_{p \in P} g(p), \sum_{p \in P} d(p) \right\}.$$

We then assign the cheapest cable to $\{v, w\}$ that has sufficient capacity and complies with the bounds above, i.e., we have for the final assignment a and all $vw \in E$

$$u_{\text{drop}}(vw) \leq u_{\text{drop}}^{\max}(\{v, w\}),$$
$$p_{\text{loss}}(vw) \leq p_{\text{loss}}^{\max}(\{v, w\}).$$

Since these requirements only strengthen the constraints in Equations (5.2) to (5.4), the cable assignment a is feasible. By Lemma 5.4 the time to compute the cable assignment is linear in the number of vertices.

Lemma 5.5. *The heuristic computes a feasible cable assignment in $\mathcal{O}(|V|)$ time.*

This cable assignment may be improved upon by iterating over the edges and greedily replacing the currently selected cable by a cable of the cheapest cable type that suffices to satisfy all constraints. We distinguish between two variants based on the order in which we iterate over the edges. Since the cost of an edge is proportional to its length, we expect greater cost saving when we can select a

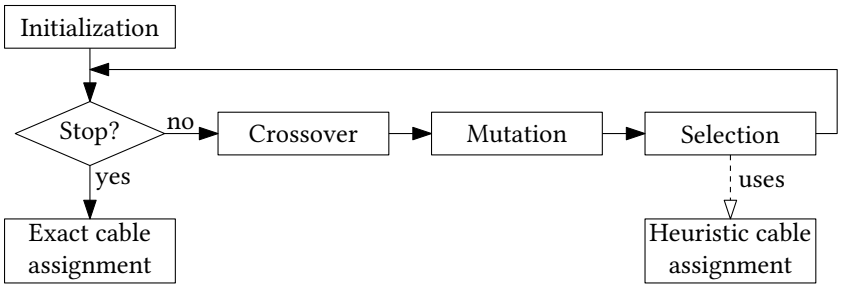


Figure 5.1: A flow chart for the hybrid genetic algorithm. The loop in the top represents the genetic algorithm that is responsible for computing topologies. The two boxes in the bottom represent cable assignment algorithms.

cheaper cable on a longer edge. Therefore, we consider the edges in the order from the longest to the shortest. The running time of the greedy improvement phase is dominated by computing the maximum generator-consumer-paths. In a topology with n_G generators and n_C consumers, there are at most $n_G \cdot n_C$ such paths. While this value may be quadratic in the total number of points, in typical microgrids there are only few generators. Thus, n_G may be thought of as constant. The length of each generator-consumer-path is linear in the number of edges. Hence, we obtain the following running time bounds for the greedy improvement.

Lemma 5.6. *The heuristic with greedy improvement computes a feasible cable assignment of a topology with n_G generators, n_C consumers, and n vertices in total in $\mathcal{O}(n_G \cdot n_C \cdot n) \subseteq \mathcal{O}(n^3)$ time.*

5.5.3 Summary of Hybrid Genetic Algorithm

Figure 5.1 summarizes the working of the hybrid genetic algorithm. In the main loop of the algorithm the topology is optimized using a genetic algorithm (see Section 5.5.1 for details). To assess the quality of the topologies computed in this process, suitable cable assignments are needed. The computation of these is delegated to one of the cable assignment algorithms in Section 5.5.2. In principle, we may use any of the three algorithms presented above (MILP, heuristic, heuristic with greedy improvement). However, we need to compute a cable assignment once per topology and iteration. It is therefore computationally infeasible to solve the MILP every time we need a cable assignment. We thus only consider the two heuristic cable assignment methods for use during the execution genetic algorithm. However, we do solve the MILP once at the end for the best topology found by the genetic algorithm.

Table 5.1: The base cable types we use in the evaluation [Kra20].

cross section [mm]	cost [USD m^{-1}]	capacity [A]	resistivity [Ω km^{-1}]
16	2.5	66	1.91
35	5.0	132	0.87
70	8.0	205	0.44
95	11.18	245	0.32

5.6 Evaluation

We implemented the algorithms described in the previous section in Rust using a modified version of genevo[3] as a framework for our genetic algorithm. We parallelized the application of the genetic operators by using rayon[4]. The experiments were performed on a SuperMicro H8QG6 Server with four 12-core AMD Opteron 6172 processors clocked at 2.1 GHz with 256 GB RAM running OpenSUSE Leap 15.3.

Due to a lack of suitable input instances, we resorted to adapting instances for the related Euclidean Minimum Steiner Tree problem, which were supplied in the DIMACS 11 implementation challenge [DS14]. These benchmark instances come in sets of different sizes. In our evaluation we chose the sets of sizes 10, 20, 50 and 100. We randomly selected between 5 % and 20 % of the points as generators with a maximum generation of 80 kW. All other points are consumers with a maximum consumption of either 4 kW (with 50 % probability), 10 kW (30 % probability), or a uniformly randomly chosen value in $[15, 75]$ (20 % probability). This distribution was chosen to resemble a collection of many households, some smaller and few larger workshops. We assume a cost of 180 USD per pole and a maximum distance between two poles of 50 m. Each distribution point incurs a fixed cost of $\alpha = 100$ USD plus $\beta = 200$ USD per connection and an additional $\gamma = 30$ USD kW^{-1}. The properties of the cables are given in Table 5.1. In addition, we assume that multiple cables of the same type may be placed in parallel to each other. We further set $\alpha_{\text{loss}} = \alpha_{\text{drop}} = 0.1$ and assume $\cos\varphi = 0.8$. We refer to the benchmark set with $k \in \{10, 20, 50, 100\}$ points by $\mathcal{B}_k$.

5.6.1 Cable Assignment

In this section we analyze the quality of the cable assignment heuristics compared to the optimal assignment. We used the points from the first 50 instances in our benchmark sets $\mathcal{B}_k$ and evaluated the heuristic and its greedy improvement. For

[3] https://github.com/innoave/genevo

[4] https://github.com/rayon-rs/rayon

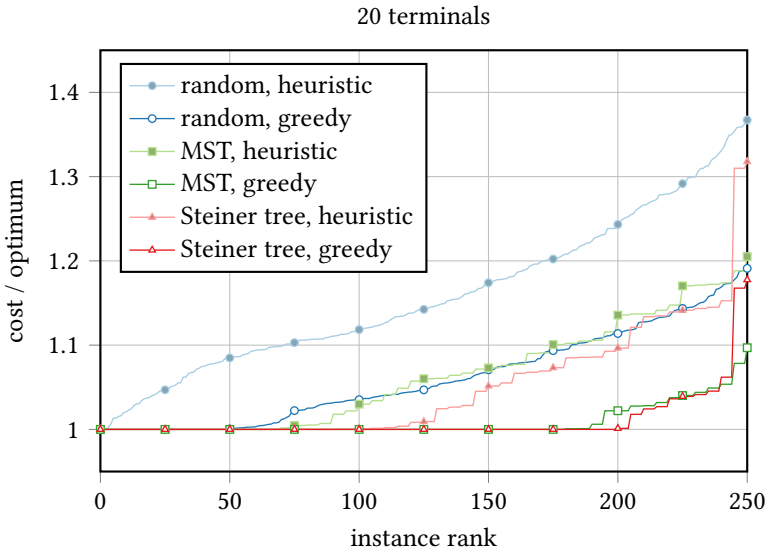


Figure 5.2: Distribution of the cost of the heuristic cable assignments relative to the optimal assignment on different generated topologies on instances from $\mathcal{B}_{20}$. Markers are placed every 25 runs corresponding to 5 instances repeated 5 times each.

each instance we used three different topologies for cable assignment: the minimum spanning tree, the minimum Steiner tree, and a random spanning tree consisting of edges in the Delaunay triangulation of the terminals. We compared the results to the minimum cost cable assignment we obtained by solving the MILP using Gurobi [Gur22]. We repeated this procedure 5 times per instance.

We find that in all test instances from $\mathcal{B}_{50}$ and $\mathcal{B}_{100}$ the heuristic always yielded the minimum cost cable assignment on all three topologies. On the test instances from $\mathcal{B}_{10}$ the heuristic was optimal on all minimum spanning tree and minimum Steiner tree topologies. The results were optimal in 80 % of all random topologies with the most expensive being around 30 % more expensive than the optimum. Using the greedy improvement this number increased further to above 90 % with the most expensive being 10 % more expensive than the optimum. The heuristic did not perform as well on $\mathcal{B}_{20}$, the results of which are presented in Figure 5.2. While there are still instances where the heuristic yielded the optimum result, it performed particularly bad on the random spanning trees. There, the base heuristic found the optimum solution in only 3 out of all 250 runs. Using the greedy improvement could, however, increase this

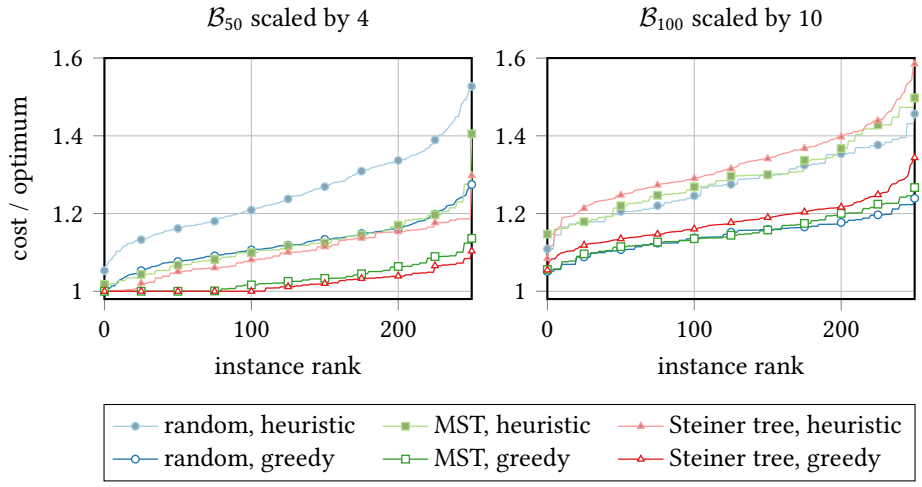


Figure 5.3: Distribution of the cost of the heuristic cable assignment relative to the optimal assignment. The coordinates in instances of $\mathcal{B}_{50}$ and $\mathcal{B}_{100}$ are scaled to match edge length distribution of $\mathcal{B}_{20}$.

to 49 runs or 19.6 %. With all three topology variants we see a large decrease in the final cost when using the greedy improvement phase.

This behavior does not seem to depend on the instance size but on the distribution of edge lengths. We scaled the coordinates of the instances in $\mathcal{B}_{20}$ by 0.25 and found that this resulted in the heuristic always finding the optimum solution. Similarly, we scaled $\mathcal{B}_{50}$ and $\mathcal{B}_{100}$ by 4 and 10, respectively, and got results similar to the ones in Figure 5.2; see Figure 5.3. These scaling factors were chosen to resemble the differences in average edge length among the instances of the different benchmark sets. The advantage of the greedy improvement becomes even more pronounced on the scaled version of $\mathcal{B}_{100}$, where a clear gap between the heuristic cable assignment with and without greedy improvement can be seen in Figure 5.3.

With fewer terminals than $\mathcal{B}_{50}$ and $\mathcal{B}_{100}$, but covering the same area, the average edge length in $\mathcal{B}_{20}$ is higher than in the other two instances. The longer the edges become on average, the more the constraints for the voltage drops and line losses become relevant. But these constraints are non-local, and the heuristic complies with stricter, localized versions of them. This is the reason why the heuristic may fail to find optimal cable assignments. In contrast, the capacity constraints are already local and are handled optimally by the heuristic. Moreover, they are independent of the edge lengths.

Table 5.2: Average running times of each cable assignment method on different benchmark sets $\mathcal{B}_k$.

	mean running time (µs)		
k	heuristic	greedy	gurobi
10	27	53	58 345
20	62	108	223 979
50	120	332	908 331
100	220	1 094	5 070 330

In terms of running time the heuristic performed best, closely followed by the heuristic with greedy improvement. The mean running times of the algorithms are given in Table 5.2. For the smaller instances adding the greedy improvement phase roughly doubles the running time. But we can also clearly see the super-linear growth of the running time for the greedy improvement that is predicted by the theoretical analysis in Lemma 5.6. The long running time of Gurobi compared to the heuristic renders it less suitable for the evaluation phase of a genetic algorithm and is the reason for our use of a heuristic.

5.6.2 Selecting Parameters for the Hybrid Genetic Algorithm

The hybrid genetic algorithm has several parameters that may influence its performance. More specifically, we have the population size s and the mutation probabilities in the close endpoint mutation π_e and in the Steiner points mutation operator π_s. To find good values for these parameters we evaluated the result of the algorithm with different parameter choices. We call a choice of the parameters a *configuration*. Based on preliminary experiments we selected $s \in \{5000, 10\,000, 15\,000, 20\,000, 25\,000\}$ and $\pi_e, \pi_s \in \{0.01, 0.02, 0.03, 0.04, 0.05, 0.075\}$. To limit the amount of combinations, we always set $\pi_e = \pi_s$. We ran each combination of parameters on 45 randomly selected instances, 15 each from $\mathcal{B}_{20}$, $\mathcal{B}_{50}$, and $\mathcal{B}_{100}$; to limit the running time we skipped $\mathcal{B}_{10}$ in our formal study since the smaller instances tend to be easier, and they depend less on the choice of the parameters. Moreover, we test both choices of the cable assignment (heuristic with and without greedy improvement).

We first determine a suitable value for the mutation rate. Table 5.3 shows for each pair of mutation rates π_1 and π_2 in which percentage of the runs on $\mathcal{B}_{20}$ and $\mathcal{B}_{50}$ the algorithm with mutation rate π_1 obtained a result that is at least as good as with mutation rate π_2. According to these results a rate of 0.05 gives at least as good results on a majority of the instances compared with all other choices of mutation

Table 5.3: Comparison of different mutation rates on $\mathcal{B}_{20}$ and $\mathcal{B}_{50}$. A value in row π_1 and column π_2 indicates the percentage of instances on which the genetic algorithm performed at least as good as with mutation rate π_1 than with rate π_2.

π	0.01	0.02	0.03	0.04	0.05	0.075
0.01	–	43.8	41.5	40.2	39.2	44.2
0.02	68.2	–	57.1	55.6	54.6	57.4
0.03	69.1	56.0	–	58.5	54.0	58.5
0.04	69.5	58.2	57.0	–	54.8	61.0
0.05	70.4	58.5	60.8	60.8	–	62.8
0.075	65.0	53.6	56.3	56.9	55.8	–

Table 5.4: Comparison of different mutation rates on $\mathcal{B}_{100}$. A value in row π_1 and column π_2 indicates the percentage of instances on which the genetic algorithm performed at least as good as with mutation rate π_1 than with rate π_2.

π	0.01	0.02	0.03	0.04	0.05	0.075
0.01	–	36.7	31.3	39.3	48.0	66.7
0.02	63.3	–	49.3	55.3	59.3	78.0
0.03	68.7	50.7	–	57.3	65.3	85.3
0.04	60.7	44.7	42.7	–	60.0	82.0
0.05	52.0	40.7	34.7	40.0	–	74.0
0.075	33.3	22.2	14.7	18.0	26.0	–

rates. The same is true for $\mathcal{B}_{20}$ and $\mathcal{B}_{50}$ individually. For $\mathcal{B}_{100}$ the results change (see Table 5.4), and a smaller mutation rate of 0.03 performs better. This makes sense since the mutation rate describes the probability that any individual edge or Steiner point is mutated. As the instances become larger, the probability that some mutation occurs increases as well. If there are many changes at once, some bad changes may overpower the good ones. This may cause the good changes to be discarded even though they would improve the topology.

Next, we compare the results with respect to the population sizes. Table 5.5 and Table 5.6 present a comparison of the population sizes similar to the tables for the mutation rates. On $\mathcal{B}_{20}$ and $\mathcal{B}_{50}$ the results are not as clear as for the mutation rates, but a population size of 10 000 seems to be the best choice. In contrast, on $\mathcal{B}_{100}$ the smaller population size of 5000 is a clear winner. This is likely because a smaller population size allows for a larger number of iterations. For larger instances, the

Table 5.5: Comparison of different population sizes on $\mathcal{B}_{20}$ and $\mathcal{B}_{50}$. A value in row s_1 and column s_2 indicates the percentage of instances on which the genetic algorithm performed at least as good as with mutation rate s_1 than with rate s_2.

s	5000	10 000	15 000	20 000	25 000
5000	–	56.6	62.5	63.9	70.2
10 000	56.2	–	66.2	68.1	73.5
15 000	50.8	47.3	–	62.8	70.4
20 000	49.8	44.6	52.3	–	70.8
25 000	42.6	40.1	44.7	42.8	–

Table 5.6: Comparison of different population sizes on $\mathcal{B}_{100}$. A value in row s_1 and column s_2 indicates the percentage of instances on which the genetic algorithm performed at least as good as with mutation rate s_1 than with rate s_2.

s	5000	10 000	15 000	20 000	25 000
5000	–	92.2	99.4	100.0	100.0
10 000	7.8	–	90.0	98.3	100.0
15 000	0.6	10.0	–	91.7	99.4
20 000	0.0	1.7	8.3	–	85.6
25 000	0.0	0.0	0.6	14.4	–

number of iterations is already lower than for smaller instances since each iteration is slower. Hence, the larger number of iterations is more important here than having a larger and more heterogeneous population.

5.6.3 Evaluation of the Solution Quality

In this section we analyze the quality of the results produced by the genetic algorithm. We aim to answer two questions. First, how good are the topologies produced by the genetic algorithms? And second, how consistent is the output of the genetic algorithms?

We start by tackling the first question. To this end we randomly selected 50 instances each from $\mathcal{B}_{10}$, $\mathcal{B}_{20}$, $\mathcal{B}_{50}$, and $\mathcal{B}_{100}$. Note that we chose these instances independently from the instances used in Section 5.6.2 to determine the best configurations. This was done to prevent giving the genetic algorithms an unfair advantage, which would happen if we assessed the quality of the topologies and determined

the best configuration on the same instances. We ran both variants of our hybrid genetic algorithm (`ga_heuristic` with the heuristic cable assignment and `ga_greedy` with the additional greedy improvement of the cable assignment) ten times on each instance for 60 s. A (relatively short) running time was selected, since the cable layout algorithm is meant to form a part of a larger algorithmic framework that addresses the full microgrid design problem. In the hybrid genetic algorithm the final cable assignment was found by solving the MILP formulation with Gurobi [Gur22].

As a baseline we compare the results of the genetic algorithms with the results of a straight-forward adaptation of the *Triangulation Algorithm* by Kahveci et al. [KOPS16]. Initially, the topology is a minimum spanning tree on the terminals. Then, triangles where two sides coincide with edges and the third side does not are considered iteratively. It is then checked whether using the third side instead of one of the others or introducing a Steiner point on an edge improves the solution. If it does, the change is applied greedily, and the next triangle is considered. In base form this heuristic is not designed to handle multiple cable types. But a straightforward way to include them is by computing an (ideally optimal) cable assignment when evaluating whether to modify the topology. We implemented two versions: one where optimal cable assignments are computed by solving the MILP, which we call `tra`, and one where the cable assignment is determined by our heuristic, which we call `tra_fast`. In both variants the final cable assignment is determined by solving the MILP.

In addition, we compare the results to the optimal cable assignments (computed by solving the MILP formulation) on the minimum spanning tree (`mst`) and the minimum Steiner tree (`steiner`). Both are solutions to geometric optimization problems that are closely related to the Microgrid Cable Layout problem. Moreover, there are approaches to similar microgrid design problems that use (or start with) the minimum spanning tree [KOPS16, LH00] and approaches introducing Steiner points [KOPS16, CCEM20]. We used the software library *geosteiner*[5] to compute minimum Steiner trees.

Figure 5.4 shows the results of the comparison of the algorithms. The plot on top includes all the algorithms mentioned above; the plot in the bottom zooms in to the area below 1.0 in order to show the results for `ga_heuristic` and `ga_greedy` more clearly. As different instances have different costs, we normalize the values by the cost of the optimal cable assignment on the topology computed by `tra`. Each line in the plot corresponds to one algorithm, and each point describes the median cost of that algorithm over the ten runs divided by the costs of the minimum Steiner tree based layout. The instances are sorted by increasing values independently for all lines. We use the median cost as this is a good indicator for a typical result of the algorithm. For example, the line for `ga_greedy` contains the point $(100, 0.475)$; this means that on 100 instances the costs of the median result of `ga_greedy` are at most 47.5 % of the costs of the `tra`-based layout.

[5] `http://www.geosteiner.com/`

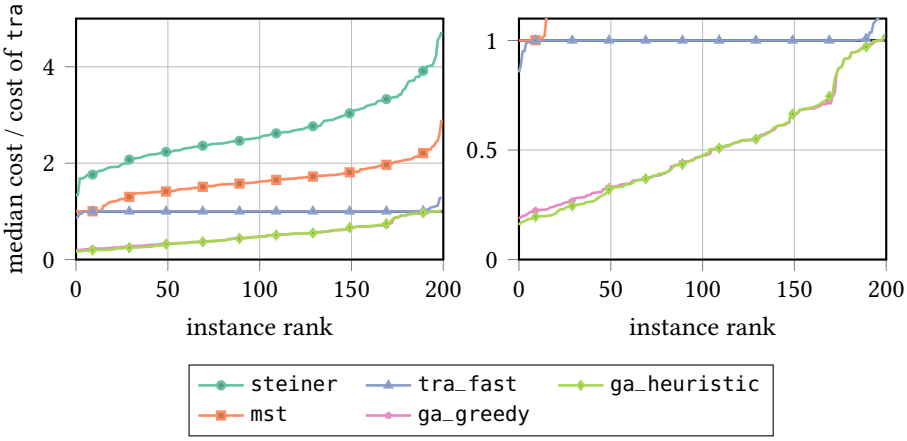


Figure 5.4: The median costs of the topologies computed by the genetic algorithms (`ga_greedy`, `ga_heuristic`) relative to the costs of the topologies computed by `tra`. As further reference, the relative costs of `mst`, `steiner`, and `tra_fast` is shown as well. The right plot is a zoomed in version of the left plot to show the lines for the genetic algorithm variants more clearly.

We can clearly see that all values of `steiner` are above 1.0. This means that `tra` always found a better topology than the minimum Steiner tree. In fact, a more detailed analysis of the data reveals that `steiner` yielded the worst topology on every single instance. This can likely be explained by the fact that in our model Steiner points incur costs and the minimum Steiner trees include many Steiner points. For `mst` all values are at least 1.0 and on all but 7 instances the values are larger than 1.0. As both `steiner` and `mst` perform worse than the baseline `tra`, we ignore them in the remainder of the evaluation section.

Both variants of the genetic algorithms perform better than `tra` on almost all instances. In Table 5.7 the columns labeled "median" correspond to the values plotted in Figure 5.4. The values indicate on how many instances the median value of the genetic algorithm over the ten runs per instance lies within the specified interval. The columns "min" and "max" correspond to the minimum and maximum values over the runs, respectively. We see that there is only one instance where the worst run of `ga_greedy` yielded a worse topology than `tra`, and two where the costs of the topologies were equal. For `ga_heuristic` the picture is similar: there are two instances with the same costs and two instances with larger costs than `tra`. Presented differently, on the vast majority of 98.5 % and 98.0 % of the instances, `ga_greedy` and

Table 5.7: The number of instances on which the relative costs of the topolgies computed by the genetic algorithms lies within the given intervals. For each instance the minimum, median, and maximum costs over the ten runs are shown for both variants of the genetic algorithm.

	`ga_greedy`			`ga_heuristic`		
	min	median	max	min	median	max
(0.10, 0.25]	29	23	8	38	32	28
(0.25, 0.50]	82	82	92	69	72	68
(0.50, 1.00)	87	93	97	90	92	100
1.00	2	2	2	2	2	2
(1.00, 1.20]	0	0	1	1	2	2

`ga_heuristic` found better solutions than `tra`.

The median costs of `ga_greedy` (`ga_heuristic`) are at most half the costs of `tra` on 52.5 % (52.0 %) of the instances and at most a quarter on 11.5 % (16 %) of the instances. In other words, on more than half the instances both variants of the genetic algorithm were able to compute topologies that cost at most half as much as the topologies computed by `tra`.

In general, the two variants of the genetic algorithm have a very similar performance. This is somewhat surprising as the greedy improvement phase is able to improve the heuristic cable assignment considerably; see Section 5.6.1. Note however that we always compute the optimal cable assignment via the MILP formulation for the final topology. Otherwise, we would see the same gap between the two variants as in Section 5.6.1.

In fact, on $\mathcal{B}_{100}$, which contains most of the instances with the largest improvement, `ga_heuristic` even produces better results than `ga_greedy` (on average 11.7 % better). This can be explained by the fact that the greedy improvement of the cable assignment increases the time needed for one iteration so that `ga_heuristic` is able to perform more iterations. On the other three benchmark sets, however, `ga_greedy` is on average slightly better than `ga_heuristic`. More precisely, on average the topologies by `ga_heuristic` are 0.04 % more expensive on $\mathcal{B}_{10}$, 1.11 % on $\mathcal{B}_{20}$, and 0.28 % on $\mathcal{B}_{50}$.

A more detailed analysis also reveals that the results by the hybrid genetic algorithms are never worse than the layout based on the minimum spanning tree. There are two instances on which the minimum spanning tree is as good as the solution found by the hybrid genetic algorithm (in all runs of both variants). But on the vast majority of the instances the minimum spanning tree is not optimal, and the hybrid genetic algorithm finds better topologies.

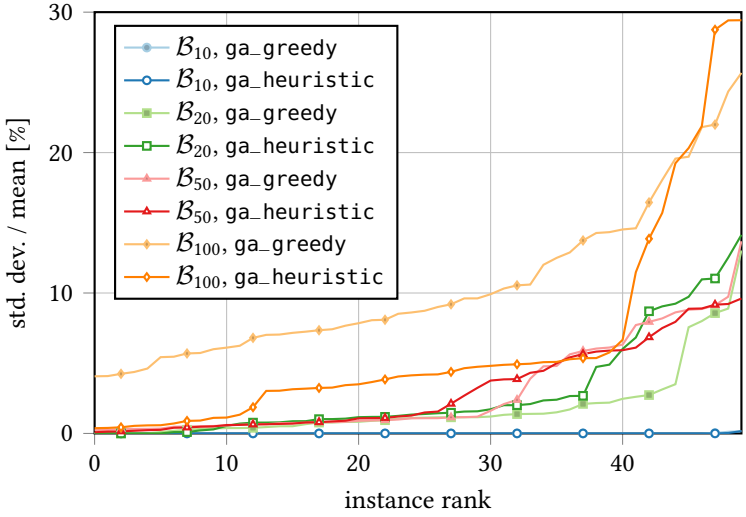
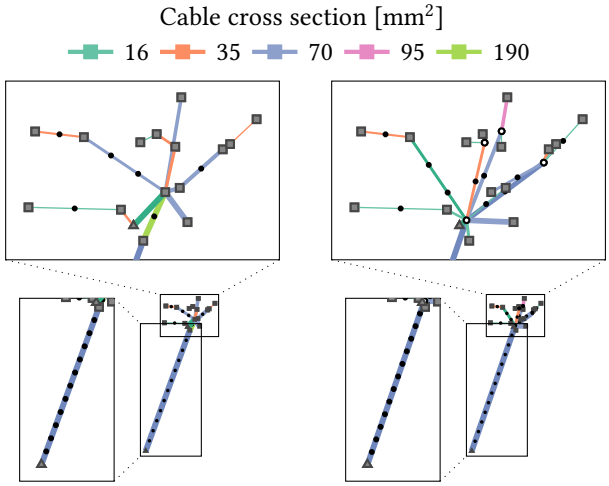


Figure 5.5: The standard deviation over ten runs of each algorithm per instance normalized by the mean cost of layouts for the instance.

Multiple runs of the genetic algorithm on the same instance may yield different results. If the costs of the results differed a lot between different runs, this would imply that the algorithm rarely finds (near-)optimal solutions. Conversely, if the results are often of similar quality, this may suggest that the solutions are (near-)optimal, in particular, because the genetic algorithm starts with different topologies every time. We compute the standard deviations of the costs of the ten runs for each instance. We normalize the values by the mean costs of the layouts for each instance. The results are shown in Figure 5.5. Each line represents the normalized standard deviations of one algorithm on the instances of one benchmark set sorted increasingly.

The comparatively large spread on $\mathcal{B}_{100}$ can likely be explained by the fact that the results are not yet optimal after 60 s; in particular, if `ga_greedy` is used. This fits to our observation that `ga_heuristic` performs slightly better on $\mathcal{B}_{100}$. On $\mathcal{B}_{10}$, $\mathcal{B}_{20}$, and $\mathcal{B}_{50}$ the relative standard deviation is small on the majority of instances; in all cases it is below 1.4 % for more than half of the instances, and on average below 3.2 %. As argued above, this may indicate that the results are often close to optimal. We observe that the relative standard deviation for `ga_greedy` (on average 0.0 % on $\mathcal{B}_{10}$, 1.9 % on $\mathcal{B}_{20}$, and 3.1 % on $\mathcal{B}_{50}$) is smaller than for `ga_heuristic` (on average 0.0 % on $\mathcal{B}_{10}$, 3.0 % on $\mathcal{B}_{20}$, and 3.1 % on $\mathcal{B}_{50}$). This means that `ga_greedy` seems to give more consistent results than `ga_heuristic`.



(a) Output of our hybrid genetic algorithm. Cost: 23 671 USD.

(b) Planned network topology. Cost: 29 649 USD.

Figure 5.6: A comparison of the topology found by our hybrid genetic algorithm and a manually planned reference topology on an instance derived from a real-world microgrid in Idjwi. In both cases the optimal cable assignment for these topologies is shown. The triangles represent generators, the squares represent consumers, and the black dots are poles.

5.6.4 Case Study: Idjwi

We evaluate the results of the hybrid genetic algorithm on an instance that is derived from a real-world microgrid in the Democratic Republic of the Congo [Tec19]. The microgrid is situated on the island of Idjwi in Lake Kivu and powers a small industrial campus. A schematic description as well as an analysis of generation and consumption data can be found in [Luh+22]. The positions of the points in the instance resemble the real-world locations of the generators and consumers. We deviate from the actual site by including a solar plant, which has not been built yet, and excluding the diesel generator. In total, we have two generators and 15 consumers. The grid uses 400 V AC line voltage and is operated at 50 Hz. We assume a power factor of $\cos\varphi = 0.8$ and limit the voltage drops and line losses to within a factor of $\alpha_{\text{drop}} = \alpha_{\text{loss}} = 0.1$.

Figure 5.6 compares the solution of our hybrid genetic algorithm (with mutation probabilities set to $\pi = 0.05$) with a manually planned reference topoglogy. The cable assignments are the optimal ones according to our model. The two topologies are

in general quite similar. They both include a long path from the generator in the bottom left to a central point in the top right. The location of the central point differs slightly in the two topologies. From this point the consumers are connected mostly along paths; in the reference solution some paths branch to two consumers.

The costs of our solution are 23 671 USD, which is significantly cheaper than the 29 649 USD for the reference solution. The cost decrease comes mainly from having fewer vertices with degree larger than 2, which incur a significant cost according to our model. Note, however, that the reference topology represents a network that has grown in multiple steps. Even if each step were to be optimal on its own, the resulting topology very likely would not be. Nevertheless, this shows that our hybrid genetic algorithm is able to significantly improve the topology.

5.7 Conclusion

In this work we present a hybrid genetic algorithm for the Microgrid Cable Layout problem. Within the algorithm we split this problem into two parts: finding a topology and computing a cable assignment for the topology. The former part is directly optimized by the genetic algorithm; the latter is done by a heuristic. The hybrid genetic algorithm achieves the two goals we formulated. First, it is able to find good solutions quickly, which is evidenced by its performance on a set of benchmark instances. It is able to consistently find solutions within 60 s, which are better than the layouts produced by heuristics presented in the literature for similar network design problems. Second, adding more constraints or changing the cost function is comparatively easy for a genetic algorithm. It may be sufficient to just adapt the evaluation function on which the selection of the new population is based. In that sense, the algorithmic approach is very flexible, which was the second goal.

The goals for the algorithm are formulated with a larger algorithmic framework covering all aspects of microgrid design in mind. Including the algorithm and studying the interactions with the other components of such a framework is therefore the main future task. This includes finding good locations for the generation units. Another interesting modification would be to compute cable capacities based on realistic generation and load scenarios. Load scenarios also allow proper power flow models to be used to obtain more accurate insights into the projected grid utilization. As in our algorithm cable assignment is part of the evaluation this can be implemented without major changes to other parts of the genetic algorithm. The hybrid genetic algorithm itself may be extended to include, e.g., obstacles, which restrict where the lines or poles may be placed, or more detailed cost functions for the equipment. For example, the choice of a pole and its foundation, and thus their costs, may depend on the forces they are subject to.

For the Microgrid Cable Layout problem it would be interesting to have an efficient algorithm for finding optimal solutions or at least solutions with approximation guarantees. A natural extension of the problem is to allow cycles in the network. One can then try to adapt the algorithm to this more general setting.

6 FACTS Flows

A FACTS flow is a generalization of an electrical flow, where the susceptances of the edges are not fixed. Instead, they can be chosen within a specified interval. We study the problem whether a given network admits a FACTS flow. We prove that in general this is $\mathcal{NP}$-hard to decide. For fixed demands this is already the case if the network is planar, and for adjustable demands this is even the case if the network is outerplanar. Previously, hardness results were only known for the problem of finding FACTS flows with maximum total flow value. On a more positive note, we give algorithms to compute a FACTS flow in cacti with adjustable demands in $\mathcal{O}(n^3)$ time and in partial 2-trees with fixed demands in $\mathcal{O}(n^2)$ time.

6.1 Introduction

To model the flow of power in electrical transmission and distribution networks, one may choose among different flow models. In the previous chapters we use electrical (often called DC power flow) and graph-theoretical flows, which are among the most commonly used models for transmission network expansion planning [LARM21, p. 7]. We study an extension of electrical flow, where the susceptances of the edges may be varied. This model is motivated by so called FACTS (flexible AC transmission system) devices [Hin93]. These devices are able to influence the power that is transmitted via a transmission line by changing the properties of the line. In practice there are multiple devices that may have different effects on the transmission lines [ZL99, KALT08]. Here, we apply a model of FACTS that allow the susceptances of the edges

to be modified continuously.

In the context of flow networks, these devices have been modeled as an extension of electrical flow networks [AP03, LBP15, LGV15, Mch+15, LPC20]. The definition of FACTS flows in this chapter follows the model by Lehmann et al. [LBP15]. The main difference to electrical flows is that the susceptances of the edges are not fixed, but may be chosen within specified intervals. We recall the definition of a *FACTS flow network* from Section 2.1.3. In this chapter we assume all input values to be rational.

A *FACTS flow network* consists of a directed graph $G = (V, E)$ with a *capacity function* $\text{cap}\colon E \to \mathbb{Q}_{>0}$, a *susceptance interval function* $B\colon E \to \text{Intervals}(\mathbb{R}_{>0})$, and a *demand function* $D\colon V \to \text{Intervals}(\mathbb{R})$. For brevity, we may refer to a FACTS flow network simply a *network* in this chapter. A *FACTS flow* in a network is a function $f\colon E \to \mathbb{R}$, for which there are functions $b\colon E \to \mathbb{R}$ and $\theta\colon V \to \mathbb{R}$ such that

$$|f(vw)| \leq \text{cap}(vw) \qquad \forall vw \in E, \tag{6.1}$$

$$\sum_{u\colon uv \in E} f(uv) - \sum_{w\colon vw \in E} f(vw) \in D(v) \qquad \forall v \in V, \tag{6.2}$$

$$f(vw) = b(vw) \cdot (\theta(v) - \theta(w)) \qquad \forall vw \in E, \tag{6.3}$$

$$b(vw) \in B(vw) \qquad \forall vw \in E. \tag{6.4}$$

The key difference to electrical flow networks is the ability to choose the susceptance $b(vw)$ of an edge vw within the interval $B(vw)$. Once the susceptances are chosen for all edges, the problem of computing a FACTS flow is reduced to computing an electrical flow. The goal of the FACTS Flow problem is to determine whether there is a FACTS flow in a network.

Related Work. The formulation above immediately gives a formulation of the FACTS Flow problem as non-convex quadratic program. This formulation can be linearized as an MILP [LBP15, LPC20]. It is mentioned in [LBP15] that FACTS Flow is in $\mathcal{NP}$, but the proof is only sketched. We therefore give a more detailed proof of membership in $\mathcal{NP}$ in Proposition 6.1.

A problem related to FACTS Flow is Maximum FACTS Flow. Instead of finding any FACTS flow, a FACTS flow with maximum total flow value shall be determined. This problem can be formulated as a decision problem as follows. Given $k \in \mathbb{Q}$ is there a FACTS flow with total flow value at least k? We call the latter problem the *decision variant* of Maximum FACTS Flow. It is known that the decision variant of Maximum FACTS Flow is strongly $\mathcal{NP}$-complete in general and $\mathcal{NP}$-complete on cacti [LBP15].

There are multiple works that introduce models of FACTS devices into Optimal Power Flow models [SC98, AAF00, CL01, AP03, SRS21]. A similar model to the one

we use here was introduced into an Optimal Power Flow model with linearized AC power flow and subsequently solved using a two-stage approach [SC98]. Other works extend AC Optimal Power Flow by including a specific kinds of FACTS such as thyristor controlled series capacitator (TCSC) [AP03] or static VAR compensator (SVC) [AAF00]. Several models and computation techniques for AC Optimal Power Flow with SVC were compared in a computational study [SRS21]. A genetic algorithm for AC Optimal Power Flow with multiple types of FACTS devices was presented [CL01].

It has also been proposed to include FACTS in a Transmission Network Expansion Planning problem. This may be formulated as a mathematical program and solved using Benders decomposition [LPC20]. Moreover, there are methods to assess the value of the flexibility offered by FACTS devices [BOGR11].

In the FACTS Flow problem and Optimal Power Flow with FACTS it is assumed that the positions and properties of the FACTS as described by the susceptance intervals are given as input. However, one may also seek to determine where to place FACTS. Placing a FACTS can be modeled as follows. Initially, the susceptances of all edges are fixed, i.e., each susceptance interval contains only one value. Placing a FACTS on an edge then means to enlarge the susceptance interval of the edge. The exact size of the resulting interval depends on the model that is used. It has been shown that when placing them with a susceptance interval of $[0, \infty)$ on edges[6] incident to a feedback vertex set, i.e., the edges without a FACTS form a forest, every flow is a FACTS flow [Lei+15]. A weaker result holds if the graph induced by the edges without a FACTS is a cactus: If the capacities of the edges on cycles of the cactus are sufficiently large, then for every flow f there is a FACTS flow f' such that the consumptions of the vertices in the two flows are equal, i.e., $c_f(v) = c_{f'}(v)$ for all vertices v [Mch+15]. Such a correspondence between graph-theoretical flows and FACTS flows is a strong property. One may also consider other (weaker) objectives, which require fewer FACTS to be placed. One such objective that has been studied is the minimization of overloads in certain contingencies. For this objective a greedy heuristic approach was presented [LA01]. Another objective is the improvement of the voltage stability [MHN22].

Contribution and Outline. While the complexity of Maximum FACTS Flow has been understood reasonably well, to the best of the author's knowledge the complexity of the underlying FACTS Flow problem is still unknown. We fill this gap. An overview over our results is shown in Table 6.1. We extend the hardness results for Maximum FACTS Flow to FACTS Flow in Section 6.2. More precisely, we show that FACTS Flow with fixed demands is strongly $\mathcal{NP}$-complete in general, FACTS Flow with fixed demands is $\mathcal{NP}$-complete on planar graphs, and FACTS Flow with

[6] Actually, that work considers placing flow control buses at the vertices, but this is equivalent to placing FACTS at all incident edges.

Table 6.1: Overview over the complexity of FACTS Flow.

Graph class	adjustable demands	fixed demands
Cactus	$\mathcal{O}(n^3)$ time (Thm. 6.13)	$\mathcal{O}(n^2)$ time (Thm. 6.11)
Outerplanar graph	$\mathcal{NP}$-complete (Thm. 6.6)	$\mathcal{O}(n^2)$ time (Thm. 6.11)
Partial 2-tree	$\mathcal{NP}$-complete (Thm. 6.6)	$\mathcal{O}(n^2)$ time (Thm. 6.11)
Planar graph	$\mathcal{NP}$-complete (Thm. 6.6)	$\mathcal{NP}$-complete (Cor. 6.7)
Arbitrary graph	strongly $\mathcal{NP}$-complete (Cor. 6.5)	strongly $\mathcal{NP}$-complete (Cor. 6.5)

adjustable demands is $\mathcal{NP}$-complete on outerplanar graphs. On a more positive note, we present an $\mathcal{O}(n^2)$-time algorithm for FACTS Flow with fixed demands in partial 2-trees (Section 6.3.1) and an $\mathcal{O}(n^3)$-time algorithm for FACTS Flow with adjustable demands in cacti (Section 6.3.2). We conclude with a short summary and open questions in Section 6.4.

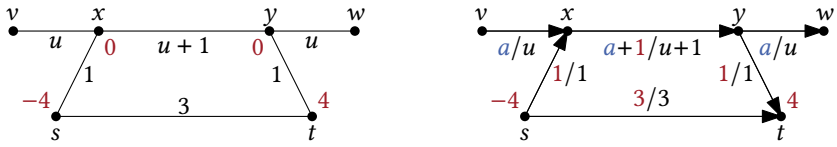
6.2 NP-Completeness

While it has been mentioned that FACTS Flow is in $\mathcal{NP}$, only a proof sketch is given [LGV15]. We fill in the gaps of the proof here.

Proposition 6.1. *FACTS Flow is in $\mathcal{NP}$.*

Proof. FACTS Flow can be formulated as an MILP, where all integer variables are binary [LBP15]. For a given variable assignment it can be checked in polynomial time whether the variable assignment is a solution, i.e., it satisfies all constraints.

However, we need to argue that a solution can be represented in a polynomial number of bits. For linear programs it is known that if all coefficients are rational and the linear program has a solution, then there is a rational solution whose size is polynomially bounded [Sch98, p. 196]. In a solution of the MILP for FACTS Flow, the binary variables can be represented with $\mathcal{O}(1)$ bits per variable. Plugging the chosen values into the MILP yields a linear program. Hence, the lemma for linear programs guarantees that the assignments to the continuous variables in some solution can be represented in a polynomial number of bits. In total, if there is a FACTS flow, then a FACTS flow can be represented in a polynomial number of bits. □



(a) The graph structure. Red values next to the vertices are demands; black values next to the edges are capacities.

(b) The structure of all FACTS flows in the gadget. The edges are labeled with $f(e)/\mathrm{cap}(e)$.

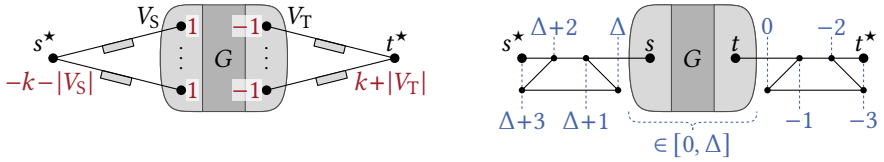
Figure 6.1: The flow bound gadget added to the edge vw.

We further prove that FACTS Flow is strongly $\mathcal{NP}$-complete by a reduction from the decision variant of Maximum FACTS Flow on basic networks. That is, in the input instances every demand interval contains 0, and no demand interval contains both positive and negative values. The problem then is to decide whether given some $k \in \mathbb{Q}$ there is a FACTS flow in the network with total flow value at least k. It is known that this problem is $\mathcal{NP}$-complete [LBP15, LGV15]. Note that the notation in [LGV15] differs from the one we use in this thesis. For easier reference we therefore restate the $\mathcal{NP}$-completeness result for Maximum FACTS Flow in the terms from Chapter 2.

Proposition 6.2 (Reformulation of Theorem 1 in [LGV15])**.** *The decision variant of Maximum FACTS Flow is strongly $\mathcal{NP}$-complete for basic networks.*

In basic networks it is easy to decide whether there is a FACTS flow. The answer is always "yes". We can simply assign a flow value of 0 to every edge and angles of 0 to every vertex. As every demand interval contains 0, this assignment satisfies the flow conservation constraints. Moreover, the capacity constraints and Kirchhoff's voltage law are satisfied as well. Thus, the hardness of finding a maximum FACTS flow does not directly imply the hardness of finding any FACTS flow. In the following we show how to reduce Maximum FACTS Flow to FACTS Flow with fixed demands. Since FACTS Flow with adjustable demands is a generalization of FACTS Flow with fixed demands, this shows that both variants of FACTS Flow are $\mathcal{NP}$-hard.

A key structure in the reduction is the *flow bound gadget*, which is illustrated in Figure 6.1. Let vw be an edge of a network $\mathcal{N}$, and let $0 \leq l \leq u$. Our goal is to add a structure that allows us to enforce a lower bound l and a upper bound u on the edge vw. Intuitively we achieve this goal by adding a structure that enforces a fixed angle difference between subdivision vertices x and y on vw. The flow along xy (and thus vx and yw) can then be controlled by choosing the susceptance interval of xy carefully.



(a) The structure of the network. Red values are demands, and the gray rectangles represent flow bound gadgets.

(b) The angles in the extension of a FACTS flow in $\mathcal{N}$ to one in $\mathcal{N}^\star$. Only one edge incident to $s^\star$ and one incident to $t^\star$ is shown.

Figure 6.2: The network $\mathcal{N}^\star$ in the reduction from Maximum FACTS Flow to FACTS Flow.

Formally, we add two subdivision vertices x and y on the edge vw. Further, we add two vertices s and t together with the edges sx, st, and yt. We set $d(x) = d(y) = 0$, $d(s) = -4$, and $d(t) = 4$. The capacities of the edges are $\mathrm{cap}(vx) = \mathrm{cap}(yw) = u$, $\mathrm{cap}(xy) = u + 1$, $\mathrm{cap}(sx) = \mathrm{cap}(yt) = 1$, and $\mathrm{cap}(st) = 3$. The susceptance intervals are $B(sx) = B(yt) = B(st) = \{1\}$, $B(xy) = [l + 1, u + 1]$, and the susceptances of the remaining edges are arbitrary and may later be chosen as desired.

Lemma 6.3. *Let $\mathcal{N}$ be a network where an edge vw has been replaced by the flow bound gadget. Then, in any FACTS flow f with angles θ in $\mathcal{N}$, it holds that $f(vx) = f(yw) \in [l, u]$ and $\theta(x) - \theta(y) = 1$.*

Proof. Let f be a FACTS flow in $\mathcal{N}$ with angles θ. By the demands of s and t and the capacities of the incident edges, we have $f(sx) = f(yt) = 1$ and $f(st) = 3$. Hence, the angle differences satisfy

$$\theta(s) - \theta(x) = 1 \qquad \theta(y) - \theta(t) = 1$$
$$\theta(s) - \theta(t) = 3 \qquad \theta(x) - \theta(y) = 1.$$

Consequently, the flow on xy satisfies $f(xy) \in B(xy) \cdot 1 = [l + 1, u + 1]$. The flow conservation at x and at y then gives $f(vx) = f(yw) \in [l, u]$. □

Given a network $\mathcal{N}$ and some $k \in \mathbb{Q}_{>0}$, we show in Theorem 6.4 how to find a network $\mathcal{N}^\star$ such that there is a FACTS flow in $\mathcal{N}$ with total flow value k if and only if there is a FACTS flow in $\mathcal{N}^\star$. Note that we consider a total flow value of exactly k in $\mathcal{N}$ rather than at least k. We argue later that this is no restriction in the setting of Maximum FACTS Flow in basic networks. Since $\mathcal{N}$ is basic, the sets of sources V_S (vertices whose demand intervals contain negative values) and sinks V_T (vertices whose demand intervals contain positive values) are disjoint. The main idea of $\mathcal{N}^\star$ is

to connect a new super source $s^\star$ to all sources and a super sink $t^\star$ to all sinks, and require a demand of k at $t^\star$; see Figure 6.2(a) for a sketch of the construction. We ensure that a FACTS flow in the restriction of the new network $\mathcal{N}^\star$ to $\mathcal{N}$ equals a FACTS flow in $\mathcal{N}$ with total flow value k, and conversely, that any such flow can be extended to a FACTS flow in $\mathcal{N}^\star$. Before presenting the details of the construction and the formal correctness proof (see Theorem 6.4) we highlight the difficulties of a naive introduction of a super source and super sink and mention how to overcome them.

For the first direction, we need to ensure that the flow on the edges between the super source and the sources (as well as between the sinks and the super sink) is within the bounds described by the demands of the sources. In particular, there must be no positive flow from a source to $V_S{}^\star$. We ensure this property by placing flow bound gadgets on the edges.

For the reverse direction, we need to ensure that every FACTS flow in $\mathcal{N}$ with total flow value k can be extended to a FACTS flow in $\mathcal{N}^\star$. But the addition of $s^\star$ and $t^\star$ introduces new cycles, along which Kirchhoff's voltage law must be satisfied. A particularly interesting case occurs when there are sources which do not generate anything in a particular flow. Suppose there are two sources s_1 and s_2, which both generate 0 units in a FACTS flow f in $\mathcal{N}$, but they have different angles. Naively adding the connections to the super source and assigning the flow to these edges would imply that there is a path from s_1 via $s^\star$ to s_2 with flow 0 on all edges. But then all vertices on this path must have the same angle. However, we want s_1 and s_2 to have the same angles as in f, which may differ from each other. To prevent this situation, we additionally send one unit of flow from $V_S{}^\star$ to each source and from each sink to $t^\star$. Then, the independence of the paths from $s^\star$ to the sources can be ensured by choosing sufficiently large susceptance intervals on the edges of the paths.

Theorem 6.4. *Let $\mathcal{N}$ be a basic network, and let $k \in \mathbb{Q}_{>0}$. There is a network $\mathcal{N}^\star$ with fixed demands such that there is a FACTS flow in $\mathcal{N}$ with total flow value exactly k if and only if there is a FACTS flow in $\mathcal{N}^\star$. Moreover, $\mathcal{N}^\star$ can be constructed from $\mathcal{N}$ in linear time.*

Proof. Let $\mathcal{N}$ be a basic network on a graph $G = (V, E)$ with demands D, capacities cap, and susceptance intervals B. As $\mathcal{N}$ is basic, we can partition V into a set of sources V_S whose demand intervals contain negative values, a set of sinks V_T whose demand intervals contain positive values, and the set $V \setminus (V_S \cup V_T)$ of intermediate vertices with demand interval $\{0\}$. We do not know the distribution of the consumptions in a FACTS flow in $\mathcal{N}$, but we do know that all sinks together shall consume k units of flow. To capture this property, we introduce a *super source* $s^\star$ and a *super sink* $t^\star$. We add $s^\star s$ for all $s \in V_S$ and $tt^\star$ for all $t \in V_T$. We set $d^\star(s^\star) = -k - |V_S|$,

$d^\star(t^\star) = k + |V_\mathrm{T}|$, $d^\star(s) = 1$ for all $s \in V_\mathrm{S}$, $d^\star(t) = -1$ for all $t \in V_\mathrm{T}$, and $d^\star(v) = 0$ for all $v \in V \setminus (V_\mathrm{S} \cup V_\mathrm{T})$.

To ensure that the flow on $s^\star s$ lies in the interval $-D(s) + 1$ for all $s \in V_\mathrm{S}$, and that the flow on $tt^\star$ lies in $D(t) + 1$ for all $t \in V_\mathrm{T}$, we place flow bound gadgets on all these edges. The resulting network is illustrated in Figure 6.2(a). To distinguish between the vertices of the gadgets on different edges, we write the original source or sink as a subscript, e.g., the vertices x_t and y_t subdivide the edge $tt^\star$. We set $\mathrm{cap}^\star(e) = \mathrm{cap}(e)$ for all $e \in E$. All other edges belong to a flow bound gadget, and we set their capacities accordingly.

Let Δ be an upper bound for the maximum angle difference between any two vertices in any FACTS flow in $\mathcal{N}$; for example, we may set

$$\Delta = \sum_{e \in E} \frac{\mathrm{cap}(e)}{\min B(e)}.$$

On the edges that belong to a flow bound gadget we set the susceptance intervals as required by the gadget, and for the remaining edges we define

$$\begin{aligned}
B^\star(s^\star x_s) &= [-\max D(s) + 1, \quad -\min D(s) + 1] && \forall s \in V_\mathrm{S},\\
B^\star(y_s s) &= \left[\frac{-\max D(s) + 1}{\Delta + 1}, \quad -\min D(s) + 1\right] && \forall s \in V_\mathrm{S},\\
B^\star(t x_t) &= \left[\frac{\min D(t) + 1}{\Delta + 1}, \quad \max D(t) + 1\right] && \forall t \in V_\mathrm{T},\\
B^\star(y_t t^\star) &= [\min D(t) + 1, \quad \max D(t) + 1] && \forall t \in V_\mathrm{T},\\
B^\star(e) &= B(e) && \forall e \in E.
\end{aligned}$$

We claim that there is a FACTS flow in the resulting network $\mathcal{N}^\star$ on the graph $G^\star$ with capacities $\mathrm{cap}^\star$ and susceptance intervals $B^\star$ if and only if there is a FACTS flow with total flow value k in $\mathcal{N}$. Suppose there is a FACTS flow $f^\star$ in $\mathcal{N}^\star$. By Lemma 6.3 the flow bound gadgets ensure that within the original network $\mathcal{N}$ the vertices in V_S and V_T act as sources and sinks as required by their demands. Of the $k + |V_\mathrm{S}|$ units of flow generated by the super source exactly k units reach the original network. Hence, $f^\star$ restricted to $\mathcal{N}$ is a FACTS flow with flow value k.

Conversely, let f be a FACTS flow in $\mathcal{N}$ with angles θ, susceptances b and flow value k. Without loss of generality, we may assume $\min\{\theta(v) \mid v \in V\} = 0$. Otherwise, we may work with θ' defined by $\theta'(v) = \theta(v) - \min\{\theta(v) \mid v \in V\}$ instead. By the definition of Δ, we have $\max\{\theta(v) \mid v \in V\} \leq \Delta$. We extend f, θ, and b to a FACTS flow $f^\star$ in $\mathcal{N}^\star$ with angles $\theta^\star$ and susceptances $b^\star$ as follows. Recall that the

consumption of a vertex v in f is denoted by $c_f(v)$. We set

$$\begin{aligned}
f^\star(s^\star x_s) = f^\star(y_s s) &= -c_f(s) + 1 && \forall s \in V_S,\\
f^\star(x_s y_s) &= -c_f(s) + 2 && \forall s \in V_S,\\
f^\star(t x_t) = f^\star(y_t t^\star) &= c_f(t) + 1 && \forall t \in V_T,\\
f^\star(x_t y_t) &= c_f(t) + 2 && \forall t \in V_T,\\
f^\star(s_v t_v) &= 3 && \forall v \in V_S \cup V_T,\\
f^\star(s_v x_v) = f^\star(y_v t_v) &= 1 && \forall v \in V_S \cup V_T,\\
f^\star(e) &= f(e) && \forall e \in E,
\end{aligned}$$

$$\begin{aligned}
\theta^\star(s^\star) = \theta^\star(s_s) &= \Delta + 3 && \forall s \in V_S,\\
\theta^\star(x_s) &= \Delta + 2 && \forall s \in V_S,\\
\theta^\star(y_s) &= \Delta + 1 && \forall s \in V_S,\\
\theta^\star(t_s) &= \Delta && \forall s \in V_S,\\
\theta^\star(s_t) &= 0 && \forall t \in V_T,\\
\theta^\star(x_t) &= -1 && \forall t \in V_T,\\
\theta^\star(y_t) &= -2 && \forall t \in V_T,\\
\theta^\star(t^\star) = \theta^\star(t_t) &= -3 && \forall t \in V_T,\\
\theta^\star(v) &= \theta(v) && \forall v \in V,
\end{aligned}$$

$$\begin{aligned}
b^\star(s^\star x_s) &= -c_f(s) + 1 && \forall s \in V_S,\\
b^\star(x_s y_s) &= -c_f(s) + 2 && \forall s \in V_S,\\
b^\star(y_s s) &= (-c_f(s) + 1)/(\Delta + 1 - \theta(s)) && \forall s \in V_S,\\
b^\star(t x_t) &= (c_f(t) + 1)/(\theta(t) + 1) && \forall t \in V_T,\\
b^\star(x_t y_t) &= c_f(t) + 2 && \forall t \in V_T,\\
b^\star(y_t t^\star) &= c_f(t) + 1 && \forall t \in V_T,\\
b^\star(s_v t_v) &= 1 && \forall v \in V_S \cup V_T,\\
b^\star(s_v x_v) = b^\star(y_v t_v) &= 1 && \forall v \in V_S \cup V_T,\\
b^\star(e) &= b(e) && \forall e \in E.
\end{aligned}$$

The angles are also illustrated in Figure 6.2(b). Direct computations confirm that the flow is conserved at every vertex and that for every edge $vw \in E^\star$ it holds that $|f^\star(vw)| \leq \mathrm{cap}^\star(vw)$, $f^\star(vw) = b^\star(vw) \cdot (\theta^\star(v) - \theta^\star(w))$, and $b^\star(vw) \in B^\star(vw)$. Hence, $f^\star$ is a FACTS flow in $\mathcal{N}^\star$.

Clearly, $\mathcal{N}^\star$ can be constructed in linear time. □

In the theorem above a FACTS flow in $\mathcal{N}^\star$ corresponds to a FACTS flow f in $\mathcal{N}$ with total flow value $\text{total}(f) = k$. However, in Maximum FACTS Flow we ask for a FACTS flow f with $\text{total}(f) \geq k$. We argue that in basic flow networks these two constraints are equivalent. Thus, with Theorem 6.4 it is possible to reduce an instance of Maximum FACTS Flow to an instance in FACTS Flow in polynomial time.

Corollary 6.5. *FACTS Flow is strongly $\mathcal{NP}$-complete even if the demands are fixed.*

Proof. By Proposition 6.2 Maximum FACTS Flow is $\mathcal{NP}$-hard on basic networks. Consider an instance of Maximum FACTS Flow on a basic network $\mathcal{N}$ with $k \in \mathbb{Q}_{\geq 0}$. That is, we ask whether there is a FACTS flow in $\mathcal{N}$ with $\text{total}(f) \geq k$. Applying the construction in Theorem 6.4 yields a network $\mathcal{N}^\star$ such that there is a FACTS flow in $\mathcal{N}^\star$ if and only if there is a FACTS flow f' in $\mathcal{N}$ with $\text{total}(f') = k$. To complete the reduction we prove that this is the case if and only if there is a FACTS flow f in $\mathcal{N}$ with $\text{total}(f) \geq k$.

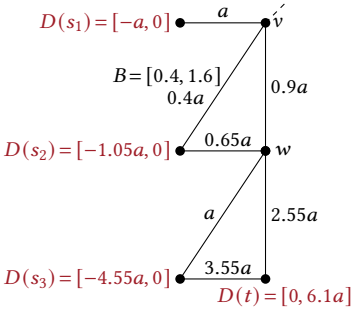
Clearly, f' satisfies $\text{total}(f') \geq k$ as well. In the other direction, let f be a FACTS flow in $\mathcal{N}$ with susceptances b and angles θ such that $\text{total}(f) \geq k$. Let $\alpha = k/\text{total}(f)$, $f' = \alpha \cdot f$, and $\theta' = \alpha \cdot \theta$. It is easy to verify that f', b, and θ' satisfy Kirchhoff's voltage law. Further, since $\alpha \leq 1$, the function f' satisfies all capacity constraints. Moreover, for the consumptions at a vertex v it holds that $c_{f'}(v) = \alpha \cdot c_f(v)$. In basic networks all demand intervals contain 0. Hence, $c_{f'}(v) \in D(v)$ for all vertices v.

The reduction in Theorem 6.4 can be computed in linear time. Thus, FACTS Flow is strongly $\mathcal{NP}$-hard. By Proposition 6.2 the decision variant of Maximum FACTS Flow belong to $\mathcal{NP}$. Therefore, FACTS Flow is in $\mathcal{NP}$ as well. □

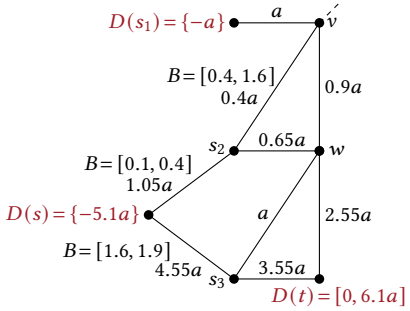
The reduction above yields graphs that are not necessarily planar. In the following we show how to adapt the reduction to show that FACTS Flow is $\mathcal{NP}$-hard even on planar graphs. It is known that Maximum FACTS Flow is $\mathcal{NP}$-hard even on cactus graphs [LGV15]. Since they are outerplanar, the introduction of a super sink as in Theorem 6.4 still yields a planar graph. However, the addition of the super source then makes the graph non-planar. The key part is to show that one common super source can be replaced by multiple "local" sources. Tho this end we consider the networks in $\mathcal{NP}$-hardness proof for Maximum FACTS Flow [LGV15] more closely.

Theorem 6.6. *FACTS Flow with adjustable demands is $\mathcal{NP}$-complete even if the underlying graph is outerplanar.*

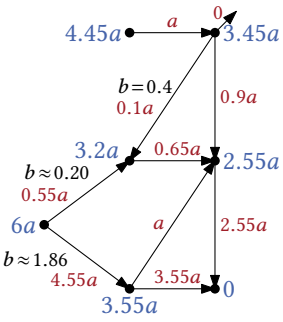
Proof. A key part in the $\mathcal{NP}$-hardness proofs for Maximum FACTS Flow [LGV15] is the existence of so-called *choice networks*; see Figure 6.3(a). A choice network has one vertex (v in the example here) designated as the *port*. The port is the only vertex via which the choice network interacts with the remainder of the network. These choice networks are designed such that they act as source that can produce either 0 or a



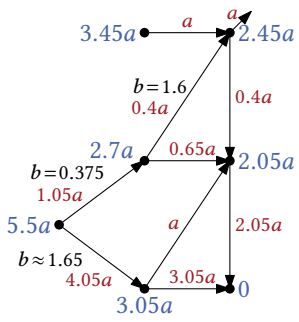
(a) Choice network for Maximum FACTS Flow.



(b) Choice network for FACTS Flow.



(c) A FACTS flow in the choice network.



(d) The other FACTS flow in the choice network.

Figure 6.3: The choice networks (a) for Maximum FACTS Flow [LGV15] and (b) for FACTS Flow. The demands are $\{0\}$ and the susceptance intervals are $\{1\}$, unless specified otherwise. In (c) and (d) the possible FACTS flows in the choice network for FACTS Flow are shown. The red values are the flow values, and the blue values are the angles.

units of flow, but no value in between, where $a > 0$ is fixed for each choice network. This is achieved by ensuring that there are two FACTS flows where the generation within the choice network is maximal (here $6.1a$ for any $a > 0$). In one flow the port has a consumption of 0, and in the other flow it has a consumption of $-a$.

Figure 6.3(b) shows how to adapt the choice network $\mathcal{N}_C$ for Maximum FACTS Flow to FACTS Flow. We fix the demand of s_1 to $\{-a\}$ as it has been shown that in any maximum FACTS flow in $\mathcal{N}_C$ the vertex s_1 has a consumption of $-a$. We

further add a super source s connected to s_2 and s_3 via edges whose capacities are the generation bounds of s_2 and s_3 in $\mathcal{N}_C$. The susceptance intervals at the new edges are $B(ss_2) = [0.1, 0.4]$ and $B(ss_3) = [1.6, 1.9]$. The demand of s is $\{-5.1a\}$, which is the sum of the consumptions of s_2 and s_3 in any maximum FACTS flow in $\mathcal{N}_C$.

We can then transfer the properties of the original choice network $\mathcal{N}_C$ to the new network $\mathcal{N}'_C$. In particular, there are exactly two FACTS flows in $\mathcal{N}'_C$: one where the consumption of v is 0 (see Figure 6.3(c)) and one where it is a (see Figure 6.3(d)). Any other FACTS flow in $\mathcal{N}'_C$ would induce another FACTS flow in the original choice network $\mathcal{N}_C$ with a generation of $6.1a$. However, this would contradict the properties of $\mathcal{N}_C$.

Note that $\mathcal{N}'_C$ is outerplanar and has only one vertex with non-fixed demand, namely t. Moreover, $D(t)$ only contains non-negative values, i.e., t is a sink.

We replace the old choice networks with the new ones in the $\mathcal{NP}$-hardness proof for Maximum FACTS Flow [LGV15]. Additionally, we fix the demands of all vertices that do not belong to a choice network to the non-zero endpoint of their demand interval. In any maximum FACTS flow in the original network these are precisely the consumptions of those vertices. Hence, the construction yields a FACTS flow network that admits a FACTS flow if and only if the original network admits a FACTS flow with the specified flow value. Moreover, the constructed network is outerplanar. □

In the networks created in the reduction all demand intervals with positive lengths contain only non-negative values, i.e., they belong to sinks. Moreover, these sinks all lie on the outer face, adding a super sink and flow bound gadgets as in the proof of Theorem 6.4 results in a planar network.

Corollary 6.7. *FACTS Flow with fixed demands is $\mathcal{NP}$-complete even if the underlying graph is planar.*

6.3 Computing FACTS Flows in Simpler Graph Classes

Having established that FACTS Flow is $\mathcal{NP}$-hard even on (outer)planar graphs if the demands are fixed (adjustable), we present two polynomial-time algorithms for FACTS Flow on simpler graph classes. We show how to compute FACTS flows (i) in partial 2-trees with fixed demands in $\mathcal{O}(n^2)$ time (see Section 6.3.1) and (ii) in cacti with adjustable demands in $\mathcal{O}(n^3)$ time (see Section 6.3.2). Both algorithms are based on explicitly computing suitable representations of feasible regions. In this section we assume all networks to be connected. Otherwise, FACTS flows can be computed for each component separately.

These regions belong to subnetworks that are connected to the remaining network via at most two vertices. Consider a network $\mathcal{N}$ with $\mathcal{N} = (G, D, \text{cap}, B)$ with two distinguished vertices: the *source terminal* s and the *sink terminal* t. Let further be $x, y \in \mathbb{R}$, and define

$$D_{x,y}(v) = \begin{cases} D(s) - x, & \text{if } v = s, \\ D(t) + y, & \text{if } v = t, \\ D(v), & \text{otherwise.} \end{cases}$$

We say that there is an (x, y, Δ)-*flow* in $\mathcal{N}$ if there is a FACTS flow in $(G, D_{x,y}, \text{cap}, B)$ with angles θ such that $\theta(s) - \theta(t) = \Delta$. Intuitively, an (x, y, Δ)-flow can be thought of as a FACTS flow in $\mathcal{N}$, where x units of flow enter at s, y units leave at t, and the angle difference between the terminals is Δ. We call the set A of all triples (x, y, Δ) for which there is an (x, y, Δ)-flow in $\mathcal{N}$ the *diagram* of $\mathcal{N}$. We use these diagrams as interfaces between different parts of the networks that are only connected via two vertices.

Observe that x and y in an (x, y, Δ)-flow are bounded by the sum of the demands of s and t and the capacities of the incident edges. Moreover, the angle difference of a single edge e is bounded by $\text{cap}(e)/\min B(e)$, and since the network is connected, Δ is bounded by the sum of all these bounds. In total, the diagrams are bounded.

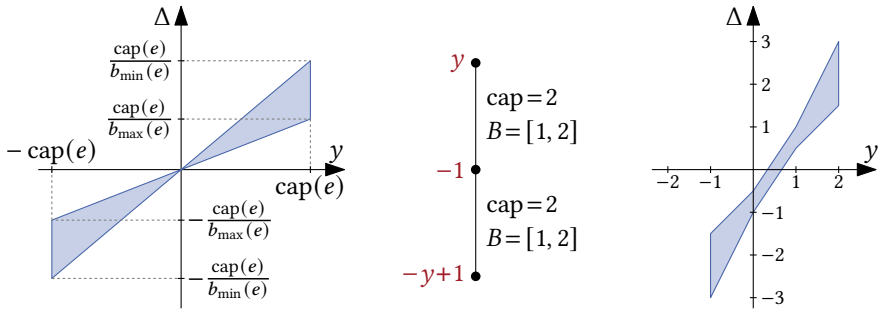
Observation 6.8. The diagrams of connected networks are bounded.

6.3.1 Fixed Demands in Partial 2-Trees

In this section we consider networks that are partial 2-trees. These are precisely the subgraphs of series-parallel graphs; see Section 2.3. Furthermore, the networks have fixed demands, i.e., there is a function d such that for all $v \in V$ it holds that $D(v) = \{d(v)\}$. Let $\mathcal{N}'$ be a subnetwork of the whole network $\mathcal{N}$ that is connected to the remainder of $\mathcal{N}$ only via the source terminal s and the sink terminal t. Let further d_{tot} be the sum of all demands in $\mathcal{N}'$. Note that in any flow f in $\mathcal{N}$, the net amount of flow that enters $\mathcal{N}'$ at s and t is precisely d_{tot}. Equivalently, this observation can be formulated in terms of (x, y, Δ)-flows. Every (x, y, Δ)-flow in $\mathcal{N}'$ satisfies $x = y + d_{\text{tot}}$. Thus, it suffices to consider the *reduced diagram* $\{(y, \Delta) \mid (x, y, \Delta) \in A\}$. Figure 6.4 shows two examples of reduced diagrams.

The main part of the algorithm deals with explicitly computing these reduced diagrams. Before describing how to do this, we introduce operations on the reduced diagrams, which are useful to formulate the algorithm. The first operation shifts a diagram A by an amount $k \in \mathbb{R}$ along the y-axis and is defined by

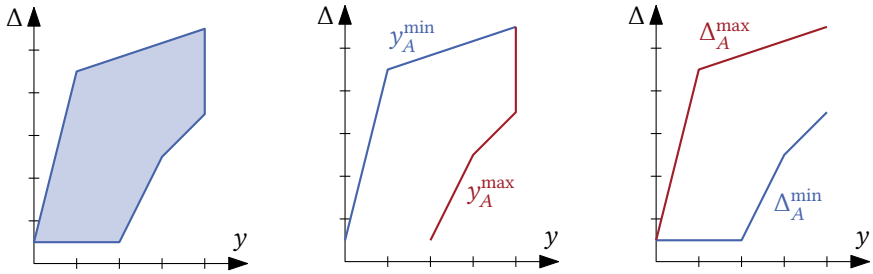
$$\text{shift}_y(A, k) = \{(y + k, \Delta) \mid (y, \Delta) \in A\}.$$



(a) The reduced diagram of a single edge e with capacity $\mathrm{cap}(e)$ and susceptance interval $[b_{\min}(e), b_{\max}(e)]$.

(b) The reduced diagram of the path on the left. Both edges have a capacity of 2 and a susceptance interval of $[1, 2]$.

Figure 6.4: Two examples of reduced diagrams.



(a) A reduced diagram A.

(b) $y_A^{\min}$ and $y_A^{\max}$.

(c) $y_A^{\min}$ and $y_A^{\max}$.

Figure 6.5: A nice reduced diagram A and its boundary functions. Note that in (b) the functions map a value of Δ to a value of y.

The other operations combine two reduced diagrams A_1 and A_2 by adding the values of one coordinate for every value of the other coordinate; see Figure 6.6 for an example.

$$A_1 +_y A_2 = \{(y_1 + y_2, \Delta) \in \mathbb{R}^2 \mid (y_1, \Delta) \in A_1,\ (y_2, \Delta) \in A_2\},$$
$$A_1 +_\Delta A_2 = \{(y, \Delta_1 + \Delta_2) \in \mathbb{R}^2 \mid (y, \Delta_1) \in A_1,\ (y, \Delta_2) \in A_2\}.$$

Let A be a reduced diagram. We say A is *ortho-convex* if $\{\Delta \mid (y^\star, \Delta) \in A\}$ and $\{y \mid (y, \Delta^\star) \in A\}$ are convex for every $y^\star, \Delta^\star \in \mathbb{R}$. Further, the *boundary functions*

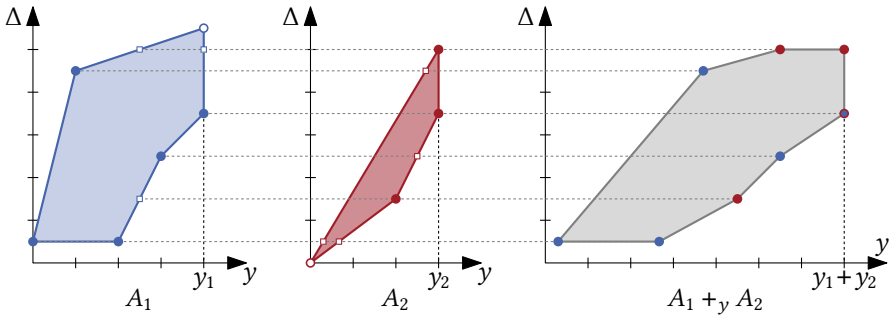


Figure 6.6: An example of $A_1 +_y A_2$.

of A are

$$y_A^{\min}(\Delta) = \min\{y \mid (y, \Delta) \in A\}, \qquad y_A^{\max}(\Delta) = \max\{y \mid (y, \Delta) \in A\},$$
$$\Delta_A^{\min}(y) = \min\{\Delta \mid (y, \Delta) \in A\}, \qquad \Delta_A^{\max}(y) = \min\{\Delta \mid (y, \Delta) \in A\},$$

where the functions are only defined if the sets on the right side are non-empty; see Figure 6.5 for an example. A reduced diagram is *nice* if it is a simple, ortho-convex polygon and all its boundary functions are monotonically increasing. We call the maximal line segments bounding a nice reduced diagram A the *sides* of A. We prove that the operations above preserve the niceness and that they can be applied in linear time. For shift_y this is clear, and for $+_y$ and $+_\Delta$ this can be achieved by adding suitable boundary functions as sketched in Figure 6.6.

Lemma 6.9. *Let A_1 and A_2 be nice reduced diagrams with p_1 and p_2 sides, respectively, and let $k \in \mathbb{R}$.*

(a) $\text{shift}_y(A_1, k)$ *is a nice reduced diagram with p_1 sides, and it can be computed in $\mathcal{O}(p_1)$ time.*

(b) $A_1 +_y A_2$ *and* $A_2 +_\Delta A_2$ *are nice reduced diagrams with at most $p_1 + p_2$ sides, and they can be computed in $\mathcal{O}(p_1 + p_2)$ time.*

Proof. The set $\text{shift}_y(A_1, k)$ is simply a translated version of A_1. Since this operation affects neither the ortho-convexity nor the monotonicity, $\text{shift}_y(A_1, k)$ is nice if and only if A_1 is nice. We can compute $\text{shift}_y(A_1, k)$ by adding k to the y-coordinates of all corners, which takes $\mathcal{O}(p_1)$ time.

The operations $+_y$ and $+_\Delta$ differ only by swapping the roles of the two coordinates. The properties of nice reduced diagrams are symmetric in the y- and the

Δ-coordinates. Hence, it suffices to prove that $A_1 +_y A_2$ has the properties we claim in the lemma.

Let $A = A_1 +_y A_2$. We first observe that for every $\Delta^\star \in \mathbb{R}$ we have

$$\{y \mid (y, \Delta^\star) \in A\} = \{y \mid (y, \Delta^\star) \in A_1\} + \{y \mid (y, \Delta^\star) \in A_2\},$$

where the + on the right is to be understood as pointwise addition. Since A_1 and A_2 are ortho-convex, the two sets on the right are convex, and thus the set on the left is convex as well. This property implies that $y_A^{\min}$ and $y_A^{\max}$ are sufficient to describe A. Moreover, it holds that

$$y_A^{\min}(\Delta) = y_{A_1}^{\min}(\Delta) + y_{A_2}^{\min}(\Delta),$$
$$y_A^{\max}(\Delta) = y_{A_1}^{\max}(\Delta) + y_{A_2}^{\max}(\Delta)$$

for those values of Δ that belong to the domains of both $y_{A_1}^{\min}$ and $y_{A_2}^{\min}$. The equalities immediately yield that $y_A^{\min}$ and $y_A^{\max}$ are monotonically increasing since the functions on the right are monotonically increasing. Furthermore, since A_1 and A_2 are simple polygons, the functions on the right side are piecewise linear and their domain is a single interval. Thus, $y_A^{\min}$ and $y_A^{\max}$ are piecewise linear and defined on a single interval as well. Hence, A is a simple polygon.

We already observed that every intersection of A with a line parallel to the y-axis is convex. To prove that A is ortho-convex, it thus remains to show that this holds for intersections with lines parallel to the Δ-axis as well. Assume for the sake of contradiction that there are $y \in \mathbb{R}$ and $\Delta_1 < \Delta_2 < \Delta_3$ such that $(y, \Delta_1) \in A$, $(y, \Delta_2) \notin A$, and $(y, \Delta_3) \in A$. But since A is a simple polygon, there is some $y_2 \in \mathbb{R}$ such that $(y_2, \Delta_2) \in A$. If $y_2 < y$, then the convexity of $\{y \mid (y, \Delta_2) \in A\}$ gives $y_A^{\max}(\Delta_2) < y$. However, $(y, \Delta_1) \in A$ implies $y \leq y_A^{\max}(\Delta_1)$, which contradicts that $y_A^{\max}$ is increasing. Similarly, $y_2 > y$ implies $y_A^{\min}(\Delta_2) > y$, and we have $y \geq y_A^{\min}(\Delta_3)$, which contradicts that $y_A^{\min}$ is increasing.

To prove the monotonicity of $\Delta_A^{\min}$ assume that there were $y_1 < y_2$ such that $\Delta_A^{\min}(y_1) > \Delta_A^{\min}(y_2)$. Let $\Delta_1 = \Delta_A^{\min}(y_1)$ and $\Delta_2 = \Delta_A^{\min}(y_2)$; see Figure 6.7. By the definition of $y_A^{\min}$ we have $y_A^{\min}(\Delta_1) \leq y_1$ and $y_A^{\min}(\Delta_2) \leq y_2$. We claim that $y_1 < y_A^{\min}(\Delta_2)$. Otherwise, the ortho-convexity of A would imply $(y_1, \Delta_2) \in A$. But then we have $\Delta_1 = \Delta_A^{\min}(y_1) \leq \Delta_2 < \Delta_1$, which is a contradiction. Thus, it holds that $y_1 < y_A^{\min}(\Delta_2)$. However, we have $y_A^{\min}(\Delta_1) \leq y_1 < y_A^{\min}(\Delta_2)$, contradicting the monotonicity of $y_A^{\min}$. The monotonicity of $\Delta_A^{\max}$ can be proved similarly. In total, we have confirmed that A is a nice reduced diagram.

Finally, we bound the number of sides of A and the time it takes to compute it. As argued above we can obtain A by adding the piecewise linear functions $y_{A_1}^{\min}$ and $y_{A_1}^{\max}$ as well as $y_{A_2}^{\min}$ and $y_{A_2}^{\max}$. Each side of A_i belongs to at most one of $y_{A_i}^{\min}$ and

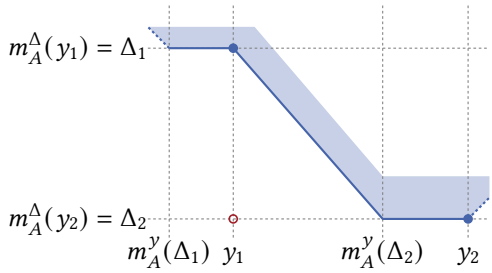


Figure 6.7: The situation in the proof of the monotonicity of $\Delta_A^{\min}$ and $\Delta_A^{\max}$ in the proof of Lemma 6.9.

$y_{A_i}^{\max}$. In fact, the representations of $y_{A_i}^{\min}$ and $y_{A_i}^{\max}$ cover all sides of A_i except the sides parallel to the y-axis. Hence, we have

$$p_i = |y_{A_i}^{\min}| + |y_{A_i}^{\max}| + k_i,$$

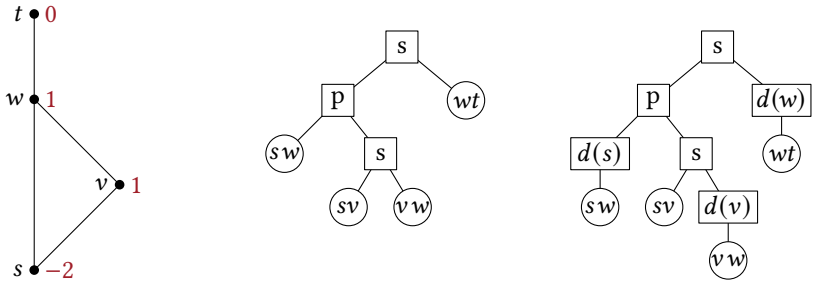
where $|\cdot|$ denotes the number of linear pieces of the functions, and k_i is the number of sides of A_i that are parallel to the y-axis.

We obtain $y_A^{\min}$ by adding $y_{A_1}^{\min}$ and $y_{A_2}^{\min}$. With the representation above, this takes $\mathcal{O}(|y_{A_1}^{\min}| + |y_{A_2}^{\min}|)$ time and produces a representation of $y_A^{\min}$ with $|y_A^{\min}| \leq |y_{A_1}^{\min}| + |y_{A_2}^{\min}| - 1$. Similarly, we obtain $y_A^{\max}$ by adding $y_{A_1}^{\max}$ and $y_{A_2}^{\max}$. Finally, we combine these two representations to get A, adding sides parallel to the y-axis if necessary. Note that we may only need to add such sides on the lines with minimum or maximum Δ-coordinate or whenever they are already present in A_1 or A_2. That is, we only add at most $k_1 + k_2 + 2$ such sides. In total, it holds for the number of sides p of A that

$$\begin{aligned} p &\leq |y_A^{\min}| + |y_A^{\max}| + k_1 + k_2 + 2 \\ &\leq \left(|y_{A_1}^{\min}| + |y_{A_2}^{\min}| - 1\right) + \left(|y_{A_1}^{\max}| + |y_{A_2}^{\max}| - 1\right) + k_1 + k_2 + 2 \\ &= p_1 + p_2. \end{aligned}$$

Computing A from the representations of A_1 and A_2 in this way takes $\mathcal{O}(p_1 + p_2)$ time. □

We use these operations to obtain the reduced diagrams of partial 2-trees. As finding a series-parallel supergraph of a partial 2-tree is possible in linear time [WC83], we focus on series-parallel graphs and describe how to extend the results to subgraphs later.



(a) The network. **(b)** The sp-tree $\mathcal{T}'$. **(c)** The spd-tree $\mathcal{T}$.

Figure 6.8: A series-parallel network and its corresponding sp-tree and spd-tree. The red numbers next to the vertices indicate the demand of the vertex.

Every series-parallel graph G can be described by an sp-tree $\mathcal{T}'$; see Section 2.3. It is a rooted binary tree, where leaves (*leaf-nodes*) correspond bijectively to edges of G and every inner node corresponds to the series composition (*s-node*) or parallel composition (*p-node*) of the subgraphs represented by its children [BLS99]. An example of an sp-tree is shown in Figure 6.8. In the description below we use the convention that "nodes" belong to the sp-tree whereas "vertices" belong to the graph G. Every node α corresponds to the subgraph containing the edges in the leaves below α in $\mathcal{T}'$. Each such subgraph is connected to the remainder of the graph via (at most) two vertices, the *terminals*, one of which is the *source terminal* and the other is the *sink terminal*. Without loss of generality we assume that the assignment of source and sink terminals to the subgraphs is consistent, i.e., in a parallel composition the two subgraphs have the same source terminal and the same sink terminal, and the source and sink terminals of a subgraph created by a series composition are the source terminal of the first subgraph and the sink terminal of the second, respectively. Furthermore, we assume that the edges are directed from the source to the sink terminal.

A network $\mathcal{N}$ does not only consist of the graph G, but it also contains other properties. In particular, the vertices have demands. To handle those, we introduce new nodes into the tree. To simplify the description of the nodes, we first ensure that the sink terminal of G has a demand of 0 by appending an edge to the graph if necessary. Then, for every vertex v with non-zero demand we insert a *d-node* into $\mathcal{T}'$ directly above the leaf-node of an edge that starts at v. If there are multiple such edges we may choose one of them arbitrarily. We denote the resulting tree by $\mathcal{T}$. Figure 6.8(c) shows the spd-tree of the example network. Note that the d-node for the vertex s may be placed either above the leaf-node for sw (as in the illustration) or above the leaf-node for sv.

We traverse the tree from the bottom up and compute the reduced diagrams of the subnetworks corresponding to the nodes. The subnetwork $\mathcal{N}(\alpha)$ that corresponds to a node α contains the edges whose leaf-nodes lie below α in $\mathcal{T}$, and a vertex v has demand $d(v)$ if the d-node for v lies below α in $\mathcal{T}$. Otherwise, the demand of v is 0. We denote the reduced diagram of $\mathcal{N}(\alpha)$ by $A(\alpha)$ and its demand sum by $d_{\text{tot}}(\alpha)$.

Lemma 6.10. *Let α be a node of $\mathcal{T}$.*

(a) If α is a leaf-node for an edge e, its reduced diagram $A(\alpha)$ consists of two triangular regions with corners $(0,0)$, $(\text{cap}(e), \text{cap}(e)/b_{\max}(e))$, $(\text{cap}(e), \text{cap}(e)/b_{\min}(e))$, $(-\text{cap}(e), -\text{cap}(e)/b_{\max}(e))$, and $(-\text{cap}(e), -\text{cap}(e)/b_{\min}(e))$, and $d_{\text{tot}}(\alpha) = 0$.

(b) If α is a d-node for a vertex v with child β, then $A(\alpha) = A(\beta)$ and $d_{\text{tot}}(\alpha) = d(v)$.

(c) If α is a p-node with children β and γ, then $A(\alpha) = A(\beta) +_y A(\gamma)$ and $d_{\text{tot}}(\alpha) = d_{\text{tot}}(\beta) + d_{\text{tot}}(\gamma)$.

(d) If α is an s-node with children β and γ, then $A(\alpha) = \text{shift}_y(A(\beta), -d_{\text{tot}}(\gamma)) +_\Delta A(\gamma)$ and $d_{\text{tot}}(\alpha) = d_{\text{tot}}(\beta) + d_{\text{tot}}(\gamma)$.

Proof. The reduced diagram of a leaf node α corresponding to an edge e is

$$A(\alpha) = \{(y, by) \mid b \in B(e), |y| \leq \text{cap}(e)\}.$$

It is easy to verify that this forms the polygon described in the lemma statement. Moreover, it holds that $d_{\text{tot}}(x) = 0$, since we consider the edge without any demands at its endpoints.

Let α be a d-node for a vertex v with the child β, which corresponds to the edge vw. Let $\mathcal{N}(\alpha)$ and $\mathcal{N}(\beta)$ be the networks consisting of the edge vw with and without the demand of $d(v)$ at v. There is a bijection between (x, y, Δ)-flows on $\mathcal{N}(\beta)$ and $(x+d(v), y, \Delta)$-flows in $\mathcal{N}(\alpha)$. Hence, the reduced diagrams $A(\alpha)$ and $A(\beta)$ are equal. The sum of all demands in $\mathcal{N}(\alpha)$ is $d(v)$.

Let α be a p-node with children β and γ, which correspond to two subnetworks $\mathcal{N}(\beta)$ and $\mathcal{N}(\gamma)$. These both have the same terminals v and w. Any (x, y, Δ)-flow in $\mathcal{N}(\alpha)$ can be split into a (x_1, y_1, Δ)-flow in $\mathcal{N}(\beta)$ and a (x_2, y_2, Δ)-flow in $\mathcal{N}(\gamma)$, and conversely any such two flows in the two subnetworks can be combined to a flow in $\mathcal{N}(\alpha)$. Hence, we have

$$\begin{aligned} A(\alpha) &= \{(y_1 + y_2, \Delta) \mid (y_1, \Delta) \in A(\beta), (y_2, \Delta) \in A(\gamma)\} \\ &= A(\beta) +_y A(\gamma). \end{aligned}$$

Since the demands of the shared vertices are accounted for in at most one of $\mathcal{N}(\beta)$ and $\mathcal{N}(\gamma)$, it further holds $d_{\text{tot}}(\alpha) = d_{\text{tot}}(\beta) + d_{\text{tot}}(\gamma)$.

Let α be an s-node with children β and γ, which correspond to two subnetworks $\mathcal{N}(\beta)$ with terminals u and v and $\mathcal{N}(\gamma)$ with terminals v and w. Any (x, y, Δ)-flow in $\mathcal{N}(\alpha)$ can be decomposed into an (x, y_1, Δ_1)-flow in $\mathcal{N}(\beta)$ and an (x_2, y, Δ_2)-flow in $\mathcal{N}(\gamma)$ where $y_1 = x_2 = y + d_{\text{tot}}(\gamma)$ and $\Delta_1 + \Delta_2 = \Delta$. Note that we have $y_1 = x_2$ since the flow that leaves $\mathcal{N}(\beta)$ at v is precisely the flow that enters $\mathcal{N}(\gamma)$ at v, and we have $x_2 = y + d_{\text{tot}}(\gamma)$, since the sum of all consumptions in the flow in $\mathcal{N}(\gamma)$ is 0. Conversely, any two such flows can be combined to a flow in $\mathcal{N}(\alpha)$. Therefore, it holds that

$$\begin{aligned} A(\alpha) &= \{(y, \Delta_1 + \Delta_2) \mid (y + d_{\text{tot}}(\gamma), \Delta_1) \in A(\beta), (y, \Delta_2) \in A(\gamma)\} \\ &= \{(y, \Delta_1 + \Delta_2) \mid (y, \Delta_1) \in \text{shift}_y(A(\beta), -d_{\text{tot}}(\gamma)), (y, \Delta_2) \in A(\gamma)\} \\ &= \text{shift}_y(A(\beta), d_{\text{tot}}(\gamma)) +_\Delta A(\gamma). \end{aligned}$$

Moreover, we have $d_{\text{tot}}(\alpha) = d_{\text{tot}}(\beta) + d_{\text{tot}}(\gamma)$ as in the case above. □

Following the structure outlined above, we obtain a quadratic-time algorithm for FACTS Flow with fixed demands in partial 2-trees.

Theorem 6.11. *In partial 2-trees with fixed demands a FACTS flow can be computed in quadratic time if it exists.*

Proof. We first check whether the sum of all demands is 0. If not, there is no FACTS flow. Afterwards, we add edges with capacity 0 and susceptance 0 such that the network is series-parallel. This is possible in linear time [WC83]. The resulting network $\mathcal{N}'$ is equivalent to the original network $\mathcal{N}$ since these edges do not admit any flow, and they do not impose any new restrictions.

The sp-tree $\mathcal{T}'$ of $\mathcal{N}'$ can be computed in $\mathcal{O}(n)$ time and has $\mathcal{O}(n)$ nodes; see Section 2.3. To obtain the spd-tree $\mathcal{T}$ we add the d-nodes to $\mathcal{T}'$, which is possible in $\mathcal{O}(n)$ time as well. We then traverse $\mathcal{T}$ from the bottom-up and compute in each step the reduced diagram $A(\alpha)$ of the current node α from the reduced diagrams of its children by applying Lemma 6.10. The reduced diagram of a leaf-node is a polygon with 6 sides by Lemma 6.10(a). Lemmas 6.9 and 6.10 together imply that $A(\alpha)$ is a polygon with at most $6m_\alpha$ sides, where m_α is the number of edges of $\mathcal{N}(\alpha)$, and that $A(\alpha)$ can be computed in $\mathcal{O}(m_\alpha)$ time from the reduced diagrams of the children of α. The reduced diagram at the root node represents the reduced diagram A of $\mathcal{N}'$. In total, A can be computed in $\mathcal{O}(n^2)$ time.

Finally, there is a FACTS flow in $\mathcal{N}'$ (and thus in $\mathcal{N}$) if and only if the reduced diagram of $\mathcal{N}'$ contains a point (y, Δ) with $y = 0$. This can be checked in $\mathcal{O}(n)$ time by traversing the boundary of $A(\rho)$. To reconstruct a flow that corresponds to a

point (y, Δ), we trace how the point was constructed, which does not increase the asymptotic running time. □

6.3.2 Adjustable Demands in Cacti

Next, we turn to the general case of networks with adjustable demands. By Theorem 6.6 computing a FACTS flow in these networks is $\mathcal{NP}$-hard even if the network is outerplanar. We therefore focus on a subclass of outerplanar graphs, namely cacti. Recall from Section 2.3 that a graph is a cactus if any two simple cycles have at most one common vertex. Consequently, every biconnected component of a cactus is either a simple cycle or a single edge.

We use the following general observation that allows us to decompose the network into biconnected components and to handle them (almost) independently. Consider a network $\mathcal{N}$ on a graph G with a cut-vertex v, i.e., we can partition G into two parts G_1 and G_2 that intersect only at v. In order to simplify the description below, we assume without loss of generality that the demand at v is $\{0\}$. This can be achieved by adding a new vertex with the original demand of v connected to v via an edge of sufficient capacity and arbitrary susceptance.

To compute a FACTS flow in $\mathcal{N}$, we first compute the amounts of flow entering the subnetwork $\mathcal{N}_1$ on G_1 at v for which there is a FACTS flow in $\mathcal{N}_1$. In other words, we temporarily treat v as vertex with infinite demand (both positive and negative) and ask for which consumptions of this vertex, there is a FACTS flow in $\mathcal{N}_1$. We call the set D_{equiv} of these values the *equivalent demand* of $\mathcal{N}_1$ with *port* v. We may then compute a FACTS flow f_2 in the subnetwork $\mathcal{N}_2$ on G_2 treating v as a vertex whose demand is D_{equiv}. Combining f_2 with a FACTS flow f_1 in $\mathcal{N}_1$ that corresponds to the consumption $c_{f_2}(v)$ of v in f_2, we get a FACTS flow for the whole network $\mathcal{N}$.

The biconnected components of a cactus are single edges or cycles. The equivalent demand of an edge vw at v (as port) can be computed easily. Computing the equivalent demand of a cycle is more involved, but we prove the following lemma.

Lemma 6.12. *The equivalent demand D_{equiv} of a cycle C with port v is an interval, and it can be computed in $\mathcal{O}(|C|^3)$ time. Moreover, given some value $d \in D_{\text{equiv}}$, a FACTS flow f in C with $c_f(v) = -d$ can be computed in $\mathcal{O}(|C|^3)$ time.*

The algorithm and the proof are quite technical, and we defer them to the end of this section. Instead, let us describe how to this lemma helps to compute a FACTS flow in a cactus.

Theorem 6.13. *A FACTS flow in a cactus can be computed in $\mathcal{O}(n^3)$ time.*

Proof. As argued before, we may assume that the demand of the cut-vertices is $\{0\}$. Let $\mathcal{T}$ be the block-cut tree of the graph G on which the network $\mathcal{N}$ is defined. We

choose an arbitrary node ρ that corresponds to a cut-vertex as root. If no such vertex exists, G is either an edge or a cycle, which can be handled directly. We iteratively compute the equivalent demands of suitable subnetworks starting at the networks represented by the leaves of the tree and work our way up to the root ρ as follows.

Consider a node α of $\mathcal{T}$. If α corresponds to a cut-vertex v, then its equivalent demand simply is the sum of the equivalent demands of its children in $\mathcal{T}$. These have been determined in previous steps. Hence, the computation can be performed in $\mathcal{O}(\deg(\alpha)) = \mathcal{O}(\deg(v))$ time.

If α corresponds to an edge vw with port v, i.e., the subnetwork of the node above α contains v, the equivalent demand is $D_{\text{equiv}}(w) \cap [-\text{cap}(vw), \text{cap}(vw)]$, where $D_{\text{equiv}}(w)$ is the equivalent demand of the child of α if it has a child or $D(w)$ otherwise. Clearly, this computation is possible in constant time.

Otherwise, α corresponds to a cycle C. All cut-vertices on C except the port correspond to children of α. Thus, their equivalent demands are known, and we use these as demands of the cut-vertices in C. By Lemma 6.12 the equivalent demand of C can then be computed in $\mathcal{O}(|C|^3)$ time.

There is a FACTS flow in the network if and only if the equivalent demand at the root ρ contains 0. After computing the equivalent demands, this can be checked in constant time.

To compute a flow, we retrace the computation of the equivalent demands. That is, we traverse $\mathcal{T}$ from the root node to the leaves. For each node α, we compute the flow on the edges of the subnetwork corresponding to α as well as the consumptions at the ports of the subnetworks of the children of α. Let α be a node and d be the consumption assigned to α by its parent; initially, we have $\alpha = \rho$ and $d = 0$.

If α corresponds to a cut-vertex v, the subnetwork has no edges, and we assign consumptions of $d_1, \dots, d_k$ to the children $\beta_1, \dots, \beta_k$ of α such that $d_1 + \cdots + d_k = d$ and $d_i \in D_{\text{equiv}}(\alpha_i)$. By the computation of the equivalent demands such a distribution of d is always possible.

If α corresponds to an edge vw with port v, we set $f(vw) = d$ and assign a consumption of d to the only child β of α.

Finally, if α corresponds to a cycle C, the second part of Lemma 6.12 gives us a FACTS flow f' in C. We then assign $c_{f'}(v)$ as the consumption to each child β of α, where v is the port of β.

For each node α computing its equivalent demand and later computing the flows takes $\mathcal{O}(m_\alpha^3 + \deg_{\mathcal{T}}(\alpha))$ time, where m_α is the number of edges in the subnetwork corresponding to the node α and $\deg_{\mathcal{T}}(\alpha)$ the degree of α in $\mathcal{T}$. Since every edge belongs to at most one cycle, we obtain a total running time of $\mathcal{O}(n^3)$. □

It remains to show Lemma 6.12, i.e., how to compute FACTS flows in cycles. The algorithm we present for this in the remainder of this section is based on explicitly

computing diagrams. But unlike in the previous section, where it is sufficient to consider reduced diagrams, the adjustable demands require us to consider the full three-dimensional diagrams.

Let C be a cycle on n vertices, called in order $v_0, \dots, v_{n-1}$, where v_0 is the port. For brevity, we set $v_n = v_0$. We denote the diagram of the path P_i from v_0 to v_i excluding the demand of v_0 but including the demand of v_i by A_i; the diagram A_0 does not include the demand of v_0. We do not explicitly store the diagrams A_i but rather representations of their upper and lower hull. More precisely, we store a representation of

$$\begin{aligned} P(A_i) &= \{(x, y) \in \mathbb{R}^2 \mid \exists \Delta \in \mathbb{R} \colon (x, y, \Delta) \in A_i\}, \\ \Delta_i^{\min}(x, y) &= \min\{\Delta \mid (x, y, \Delta) \in A_i\}, \\ \Delta_i^{\max}(x, y) &= \max\{\Delta \mid (x, y, \Delta) \in A_i\}, \end{aligned}$$

where the two functions are only defined for $(x, y) \in P(A_i)$.

In a first step, we establish relevant properties of A_i, $P(A_i)$, $\Delta_i^{\min}$, and $\Delta_i^{\max}$. The first two results deal with the structure of the diagrams and can be obtained by interpolating flows in a suitable way.

Lemma 6.14. *The set $P(A_i)$ is convex for $i \in \{0, \dots, n\}$.*

Proof. Let $(x_A, y_A), (x_B, y_B) \in P(A_i)$. We need to prove that all points on the line segment between these two points, i.e., all points $(x_\xi, y_\xi) := ((1-\xi)x_A + \xi x_B,\ (1-\xi)y_A + \xi y_B)$ for $\xi \in [0, 1]$, belong to $P(A_i)$ as well. By the definition of $P(A_i)$ there are an (x_A, y_A, Δ_A)-flow f_A and an (x_B, y_B, Δ_B)-flow f_B on P_i for suitable values $\Delta_A, \Delta_B \in \mathbb{R}$. By interpolating f_A and f_B we obtain flows $f_\xi = (1-\xi)f_A + \xi f_B$ for $\xi \in [0, 1]$. Note that for $\xi \in [0, 1]$ the flow f_ξ satisfies all edge capacities. Moreover, x_ξ units of flow enter the path at v_0, and y_ξ units leave the path at v_i. Since P_i is a path, every flow, and in particular f_ξ, is a FACTS flow. Hence, there is some value $\Delta_\xi \in \mathbb{R}$ such that $(x_\xi, y_\xi, \Delta_\xi) \in A_i$, and thus $(x_\xi, y_\xi) \in P(A_i)$. □

The previous lemma is concerned with the convexity in directions within the xy-plane. The following lemma deals with lines parallel to the Δ-axis.

Lemma 6.15. *Let $i \in \{0, \dots, n\}$ and $(x, y) \in P(A_i)$. It holds that*

$$A_i \cap (\{(x, y)\} \times \mathbb{R}) = \{(x, y)\} \times [\Delta_i^{min}(x, y), \Delta_i^{max}(x, y)].$$

Proof. By the definitions of $\Delta_i^{\min}$ and $\Delta_i^{\max}$ we have $A_i \cap \{(x, y)\} \times \mathbb{R} \subseteq \{(x, y)\} \times [\Delta_i^{\min}(x, y), \Delta_i^{\max}(x, y)]$. It remains to show the inclusion in the other direction. There are FACTS flows $f^{\min}$ and $f^{\max}$ on P_i with angles $\theta^{\min}$ and $\theta^{\max}$ such that

$\theta^{\min}(v_i) = \theta^{\max}(v_i) = 0$, $\theta^{\min}(v_0) = \Delta_i^{\min}(x, y)$, and $\theta^{\max}(v_0) = \Delta_i^{\max}(x, y)$. For $\diamond \in \{\Delta^{\min}, \Delta^{\max}\}$ let $b_\diamond(v_j v_{j+1}) = f_\diamond(v_j v_{j+1})/(\theta_\diamond(v_{j+1}) - \theta_\diamond(v_j))$.

For $\xi \in [0, 1]$ we set $f_\xi = \xi f^{\max} + (1 - \xi) f^{\min}$ and $b_\xi = \xi b^{\max} + (1 - \xi) b^{\min}$. Since P_i is a path, there are angles θ_ξ such that $f_\xi(v_j v_{j+1}) = b_\xi(v_j v_{j+1}) \cdot (\theta_\xi(v_{j+1}) - \theta_\xi(v_j))$ for all j, and $\theta_\xi(v_i) = 0$. Then, f_ξ is a FACTS flow with angle difference $\theta_\xi(v_i)$ between v_0 and v_i. Hence, it is a witness for the point $(x, y, \theta_\xi(v_i))$ in A_i. Since $\theta_\xi(v_i)$ is continuous in ξ, $\theta_0 = \theta^{\min}$, and $\theta_1 = \theta^{\max}$, the mean value theorem implies that for every $\Delta \in [\Delta_i^{\min}(x, y), \Delta_i^{\max}(x, y)]$ there is a corresponding FACTS flow f_ξ. Hence, $A_i \cap \{(x, y)\} \times \mathbb{R} = \{(x, y)\} \times [\Delta_i^{\min}(x, y), \Delta_i^{\max}(x, y)]$. □

We further prove that $\Delta_i^{\min}$ and $\Delta_i^{\max}$ are monotonic.

Lemma 6.16. *Let $i \in \{0, \dots, n\}$, and let $x, y_1, y_2 \in \mathbb{R}$ such that $(x, y_1), (x, y_2) \in P(A_i)$, and $y_1 < y_2$. Then, $\Delta_i^{min}(x, y_1) \le \Delta_i^{min}(x, y_2)$ and $\Delta_i^{max}(x, y_1) \le \Delta_i^{max}(x, y_2)$.*

Similarly, let $x_1, x_2, y \in \mathbb{R}$ such that $(x_1, y), (x_2, y) \in P(A_i)$, and $x_1 < x_2$. Then, $\Delta_i^{min}(x_1, y) \le \Delta_i^{min}(x_2, y)$ and $\Delta_i^{max}(x_1, y) \le \Delta_i^{max}(x_2, y)$.

Proof. We only prove the first statement and only the part about $\Delta_i^{\max}$; the one for $\Delta_i^{\min}$ can be obtained similarly, and the second statement can be obtained by reversing the direction of the path, which switches the roles of the x- and y-coordinates.

Let f_1 be a $(x, y_1, \Delta_i^{\max}(x, y_1))$-flow in P_i with angles θ_1. Without loss of generality we may assume $\theta_1(v_i) = 0$, and thus, $\theta_1(v_0) = \Delta_i^{\max}(x, y_1)$. Our goal is to obtain a (x, y_2, Δ_2)-flow with $\Delta_2 \ge \Delta_i^{\max}(x, y_1)$, which directly implies $\Delta_i^{\max}(x, y_2) \ge \Delta_i^{\max}(x, y_1)$.

To this end, we identify the last vertex v_j of P_i that does not consume the minimum amount in f_1, i.e., $c_{f_1}(v_j) > \min D(v_j)$. Such a vertex exists, since $y_1 < y_2$ and $(x, y_2) \in P(A_i)$. We then form f' by sending additional $\alpha = \min\{c_{f_1}(v_j) - \min D(v_j), y_2 - y_1\}$ units from v_j to v_i. If $y_1 + \alpha = y_2$, then y_2 units of flow leave the path at v_i in f', and we set $f_2 = f'$. Otherwise, it holds that $y_1 + \alpha < y_2$, and we repeat this process until the first case holds. As argued for the first step that this process never fails to find a vertex v_j whose consumption is larger than its minimum demand. Thus, the procedure always finds a function f_2.

However, we still need to argue that f_2 is a (x, y_2, Δ_2)-flow for a sufficiently large value Δ_2. By construction, f_2 satisfies the flow conservation constraints at all vertices. We further claim that it observes the edge capacities. On the edges where $f_2(v_j v_{j+1}) = f_1(v_j v_{j+1})$ this is clearly the case. If the flow $f_2(v_j v_{j+1})$ on some edge $v_j v_{j+1}$ (with $0 \le j < i$) differs from $f_1(v_j v_{j+1})$, all vertices from v_{j+1} to v_i consume the minimum amount of flow by the construction of f_2. Let f_2' be a (x, y_2, Δ_2')-flow for some $\Delta_2' \in \mathbb{R}$; since $(x, y_2) \in P(A_i)$ such a flow exists. Note that to satisfy the demands of the vertices $v_{j+1}, \dots, v_i$, the amount of flow $f_2'(v_j v_{j+1})$ must be at least y_2 plus

the minimum demands of all the vertices $v_{j+1}, \dots, v_i$; a property that we use in the inequality marked with $(*)$ below. It then holds that

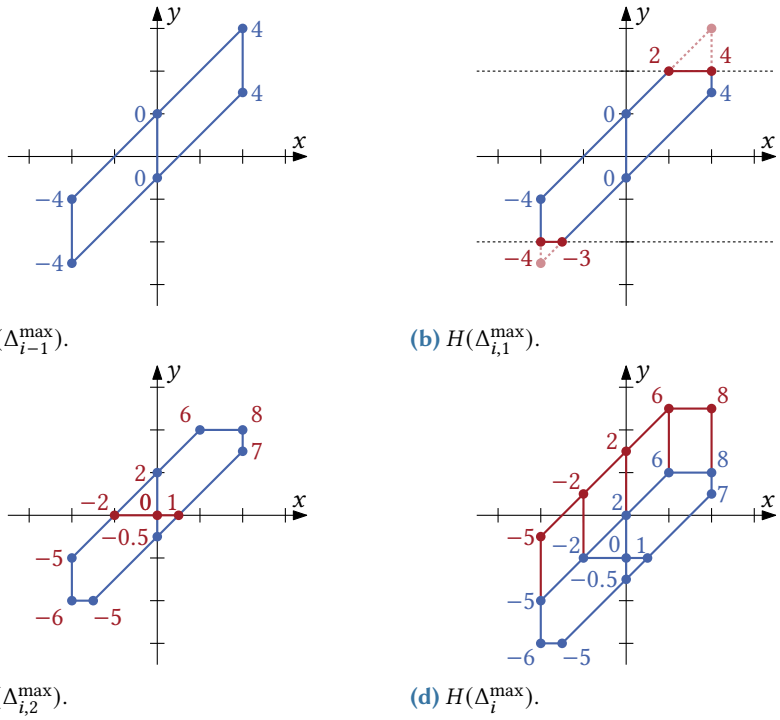
$$\begin{aligned} -\mathrm{cap}(v_j v_{j+1}) &\leq f_1(v_j v_{j+1}) \\ &\leq f_2(v_j v_{j+1}) \\ &= y_2 + \sum_{k=j+1}^{i} \min D(v_k) \\ &\overset{(*)}{\leq} f_2'(v_j v_{j+1}) \\ &\leq \mathrm{cap}(v_j v_{j+1}). \end{aligned}$$

Setting the susceptances of the edges to the same values as for f_1, we derive angles θ_2 corresponding to f_2 with $\theta_2(v_i) = 0$ and $\Delta_2 := \theta_2(v_0) \geq \theta_1(v_0) = \Delta_i^{\max}(x, y_1)$. □

Lemma 6.15 implies that $\Delta_i^{\min}$ and $\Delta_i^{\max}$ are sufficient to obtain A_i. In our algorithm we therefore store representations of $\Delta_i^{\min}$ and $\Delta_i^{\max}$. We represent $\Delta_i^{\min}$ and $\Delta_i^{\max}$ by the drawings of two planar graphs $H(\Delta_i^{\min})$ and $H(\Delta_i^{\max})$ with values assigned to the vertices. We call these values the *associated values* of the vertices. These drawings have the following properties. The vertices are assigned to coordinates, and the edges are drawn as line segments. The edges on the outer face enclose precisely $P(A_i)$. The interior of $P(A_i)$ may be subdivided by vertices and edges, but each face of the subdivision is convex. Moreover, we ensure that $\Delta_i^{\min}(x, y)$ for some $(x, y) \in P(A_i)$ can be determined from $H(\Delta_i^{\min})$ as follows. If a vertex v of $H(\Delta_i^{\min})$ lies at (x, y), then the value associated with v is precisely $\Delta_i^{\min}(x, y)$. Otherwise, the point (x, y) lies within a face of $H(\Delta_i^{\min})$. Hence, it is possible to represent (x, y) as some convex combination $\sum \xi_j(x_j, y_j)$ of the positions of vertices incident to the face. We ensure that $\Delta_i^{\min}(x, y)$ can be obtained by interpolating the associated values at the vertices in the same way, i.e., we have $\Delta_i^{\min}(x, y) = \sum \xi_j \Delta^{\min}(x_j, y_j)$. To ensure that this interpolation works, we construct $H(\Delta_i^{\min})$ in such a way that $\Delta_i^{\min}$ is an affine function when restricted to any face of $H(\Delta_i^{\min})$. The values of $\Delta_i^{\max}$ can be obtained from $H(\Delta_i^{\max})$ in the same way.

We have $A_0 = \{(x, x, 0) \mid x \in \mathbb{R}\}$, and thus $P(A_0) = \{(x, x) \mid x \in \mathbb{R}\}$ and $\Delta_0^{\min}(x, y) = \Delta_0^{\max}(x, y) = 0$. Instead of storing an infinite line, we may limit our representation to a suitably large square of side length $2N$ around the origin. Then, the representations of $\Delta_0^{\min}$ and $\Delta_0^{\max}$ are a single edge between vertices at $(-N, -N)$ and (N, N), both of which have 0 as associated values.

To obtain $H(\Delta_i^{\min})$ and $H(\Delta_i^{\max})$ from $H(\Delta_{i-1}^{\min})$ and $H(\Delta_{i-1}^{\max})$, we perform three steps. First, we handle the capacity of the additional edge $v_{i-1}v_i$. Then, we include the angle difference of the flow along $v_{i-1}v_i$. And finally, we handle the demand of v_i.



(a) $H(\Delta_{i-1}^{\max})$. **(b)** $H(\Delta_{i,1}^{\max})$.

(c) $H(\Delta_{i,2}^{\max})$. **(d)** $H(\Delta_{i}^{\max})$.

Figure 6.9: An example of the steps for building the representation of $\Delta_i^{\max}$. The edge $v_{i-1}v_i$ has a capacity of 2 and a susceptance interval of $[1, 2]$, and the demand interval of v_i is $[-0.5, 1]$. In each step, the new parts are marked in red. In (b) the capacity of $v_{i-1}v_i$ is included. In (c) the susceptance of $v_{i-1}v_i$ is included. In (d) the demand interval of v_i is included.

We denote the graphs obtained after step j for the path P_i by $H(\Delta_{i,j}^{\min})$ and $H(\Delta_{i,j}^{\max})$, the corresponding diagram by $A_{i,j}$, and its projection to the xy-plane by $P(A_{i,j})$. An example of the steps is shown in Figure 6.9.

Step 1: Capacity of $v_{i-1}v_i$. We have $A_{i,1} = \{(x, y, \Delta) \in A_{i-1} \mid |y| \leq \operatorname{cap}(v_{i-1}v_i)\}$. Hence, we obtain $H(\Delta_{i,1}^{\min})$ from $H(\Delta_{i-1}^{\min})$ by cutting the graph at the lines $y = -\operatorname{cap}(v_{i-1}v_i)$ and $y = \operatorname{cap}(v_{i-1}v_i)$, subdividing the edges at the intersections with these lines if necessary, adding edges between the vertices on the lines, and finally removing all parts outside of the strip bounded by the lines. The values associated

with the new vertices are the linear interpolations of the values at the endpoints of the old edges the vertices lie on. Similarly, we obtain $H(\Delta_{i,1}^{\max})$.

Step 2: Susceptance of $v_{i-1}v_i$. To include $B(v_{i-1}v_i)$, we cut $H(\Delta_{i,1}^{\min})$ and $H(\Delta_{i,1}^{\max})$ at the line $y = 0$, adding vertices and edges as in the previous step if necessary. The resulting graphs are $H(\Delta_{i,2}^{\min})$ and $H(\Delta_{i,2}^{\max})$. We then change the values associated with the vertices by setting

$$\Delta_{i,2}^{\min}(x, y) = \begin{cases} \Delta_{i,1}^{\min}(x, y) + y \cdot \max B(v_{i-1}v_i), & y \leq 0, \\ \Delta_{i,1}^{\min}(x, y) + y \cdot \min B(v_{i-1}v_i), & y > 0, \end{cases} \tag{6.5}$$

$$\Delta_{i,2}^{\max}(x, y) = \begin{cases} \Delta_{i,1}^{\max}(x, y) + y \cdot \min B(v_{i-1}v_i), & y \leq 0, \\ \Delta_{i,1}^{\max}(x, y) + y \cdot \max B(v_{i-1}v_i), & y > 0. \end{cases} \tag{6.6}$$

Step 3: Demand of v_i. Finally, we include the demand of v_i. We obtain $H(\Delta_i^{\max})$ from $H(\Delta_{i,2}^{\max})$ as follows. We first subtract $\max D(v_i)$ from all y-coordinates, which effectively moves the graph by $-\max D(v_i)$ units along the y-axis. Then, we add a copy of the upper boundary of $H(\Delta_{i,2}^{\max})$, i.e., those vertices and edges with maximum y-coordinate for each x-coordinate, but moved by $-\min D(v_i)$ units along the y-axis from its original position in $H(\Delta_{i,2}^{\max})$. The two copies of the upper boundary are exactly $\max D(v_i) - \min D(v_i)$ apart. We finally connect the two copies of each vertex on the upper boundary of $H(\Delta_{i,2}^{\max})$ by an edge. The values associated with the vertices stay the same during this construction. In particular, the two copies of the vertices on the bottom boundary have the same associated value. The construction for $H(\Delta_i^{\min})$ is similar, but we first move the graph by $-\min D(v_i)$ units, and then add a copy of the bottom boundary translated by $-\max D(v_i)$. The monotonicity of the boundary functions (Lemma 6.16) plays a crucial role in proving the correctness of the constructions.

Lemma 6.17. *The graphs $H(\Delta_i^{min})$ and $H(\Delta_i^{max})$ are correct representations of Δ_i^{min} and Δ_i^{max}. In particular, Δ_i^{min} and Δ_i^{max} are continuous.*

Proof. Clearly the representations of $H(\Delta_0^{\min})$ and $H(\Delta_0^{\max})$ are correct. To get from A_{i-1} to A_i, we perform three steps. In the first step we include the capacity of the edge $v_{i-1}v_i$. It holds that $A_{i,1} = \{(x, y, \Delta) \in A_{i-1} \mid |y| \leq \operatorname{cap}(v_{i-1}v_i)\}$. Thus, cutting the graphs at the lines $y = -\operatorname{cap}(v_{i-1}v_i)$ and $y = -\operatorname{cap}(v_{i-1}v_i)$ yields a correct representation of $H(\Delta_{i,1}^{\min})$ and $H(\Delta_{i,1}^{\max})$.

In the second step we include the susceptance interval of the edge $v_{i-1}v_i$. A point $(x, y, \Delta) \in A_{i,1}$ corresponds to a FACTS flow on P_i, where y units flow

along $v_{i-1}v_i$. Thus, we have

$$A_{i,2} = \{(x, y, \Delta + b \cdot y) \mid (x, y, \Delta) \in A_{i,1}, b \in B(v_{i-1}v_i)\}.$$

Hence, it holds that

$$\begin{aligned}
\Delta^{\min}_{i,2}(x, y) &= \Delta^{\min}_{i,1}(x, y) + \min\{b \cdot y \mid y \in B(v_{i-1}v_i)\} \\
&= \begin{cases} \Delta^{\min}_{i,1}(x, y) + y \cdot \max B(v_{i-1}v_i), & y \leq 0, \\ \Delta^{\min}_{i,1}(x, y) + y \cdot \min B(v_{i-1}v_i), & y > 0, \end{cases} \\
\Delta^{\max}_{i,2}(x, y) &= \Delta^{\max}_{i,1}(x, y) + \max\{b \cdot y \mid y \in B(v_{i-1}v_i)\} \\
&= \begin{cases} \Delta^{\max}_{i,1}(x, y) + y \cdot \min B(v_{i-1}v_i), & y \leq 0, \\ \Delta^{\max}_{i,1}(x, y) + y \cdot \max B(v_{i-1}v_i), & y > 0. \end{cases}
\end{aligned}$$

By construction $\Delta^{\min}_{i,1}$ is affine within each face of $H(\Delta^{\min}_{i,1})$. The only difference between $\Delta^{\min}_{i,2}$ and $\Delta^{\min}_{i,1}$ is the addition of $\min\{b \cdot y \mid y \in B(v_{i-1}v_i)\}$, which is affine within the halfplanes to either side of the line defined by $y = 0$. Splitting the graph $H(\Delta^{\min}_{i,1})$ at this line hence ensures that within each face $\Delta^{\min}_{i,2}$ are affine function. Thus, after applying the transformations in Equations (6.5) and (6.6) only to the values at the vertices, the correct values of all other points can be obtained by interpolation. A similar argument applies to $\Delta^{\max}_{i,2}$ and $H(\Delta^{\max}_{i,2})$.

In the third step we include the demand of v_i, and it holds that

$$A_i = \{(x, y + d, \Delta) \mid (x, y, \Delta) \in A_{i,2}, d \in D(v_i)\}.$$

Hence, we have

$$\Delta^{\max}_i(x, y) = \max_{d \in D(v_i)} \Delta^{\max}_{i,2}(x, y + d).$$

By Lemma 6.14 the set $P(A_i)$ is convex, and by Lemma 6.16 the functions $\Delta^{\min}_{i-1}$ and $\Delta^{\max}_{i-1}$ are increasing for fixed x. The same holds for $\Delta^{\min}_{i,2}$ and $\Delta^{\max}_{i,2}$ since the operations in the previous two steps preserve this property. Thus, in the equation above the maximum is attained for $d = \max D(v_i)$ provided that $y + \max D(v_i) \leq y_{\max}(x)$, where $y_{\max}(x) = \max\{y \mid (x, y) \in P(A_{i,2})\}$. Otherwise, the maximum is equal to $\Delta^{\max}_{i,2}(x, y_{\max}(x))$. To summarize, we have

$$\Delta^{\max}_i(x, y) = \begin{cases} \Delta^{\max}_{i,2}(x, y + \max D(v_i)), & \text{if } y + \max D(v_i) \leq y_{\max}(x), \\ \Delta^{\max}_{i,2}(x, y_{\max}(x)), & \text{otherwise.} \end{cases}$$

The copy of $H(\Delta^{\max}_{i,2})$ translated by $-\max D(v_i)$ along the y-axis corresponds to the first case of the equation above. The area between the two copies of the upper

boundary corresponds to the second case. In total, this shows that the graph $P(\Delta_i^{\max})$ is a correct representation of $\Delta_i^{\max}$. A similar argument shows that the procedure for $\Delta_i^{\min}$ is correct as well.

Moreover, since it is possible to represent $\Delta_i^{\min}$ and $\Delta_i^{\max}$ via the graphs, the two functions are continuous. □

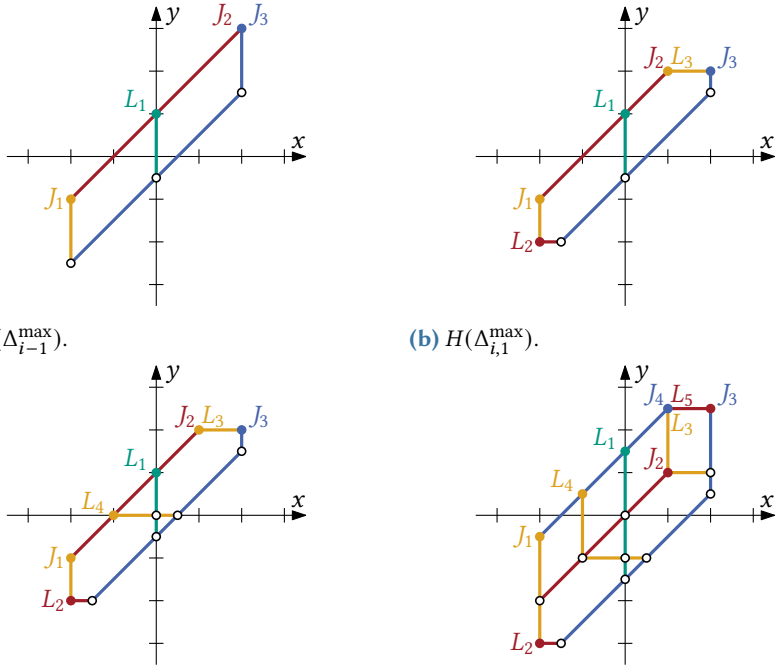
We further bound the sizes of the graphs $H(\Delta_i^{\min})$ and $H(\Delta_i^{\max})$. To this end, we cover the edges of the graphs with $\mathcal{O}(i)$ shapes such that all but a constant number of vertices per shape is located at the intersection of two shapes. Moreover, the shapes are chosen such that any two shapes intersect at most once. All in all, we get $\mathcal{O}(i^2)$ vertices and edges.

Lemma 6.18. *The graphs $H(\Delta_i^{min})$ and $H(\Delta_i^{max})$ have $\mathcal{O}(i^2)$ vertices and edges.*

Proof. We prove the bounds for $H(\Delta_i^{\max})$; the proof for $H(\Delta_i^{\min})$ is similar. By construction the angle of an edge to the x-axis is either $0°$ for *horizontal* edges, $45°$ for *diagonal* edges, or $90°$ for *vertical* edges. We further cover the edges with paths, which we call *shapes*, such that every edge belongs to exactly one shape. There are two shapes: *L-shapes* and *J-shapes*. An L-shape is a path that first traces horizontal edges (with decreasing x-coordinates) and then vertical edges (with increasing y-coordinates). A J-shape is a path that first traces diagonal edges (with increasing y-coordinate) and then vertical edges (with increasing y-coordinate). Note that we allow shapes to have edges in only one direction and even to only consist of one vertex. We call the vertex of a shape on the vertical path with the maximum y-coordinate the *upper endpoint* of the shape. In Figure 6.10 the example graphs of Figure 6.9 are covered with shapes. The upper endpoints of the shapes are marked by filled disks. Note that multiple shapes may have a common upper endpoint.

We prove in the following that $\mathcal{O}(i)$ shapes suffice to cover all edges of $H(\Delta_i^{\max})$ such that (a) all vertices on the upper boundary are upper endpoints of shapes, (b) the upper boundary, which are the points of maximum y-coordinate for each x-coordinate, is formed by at most one J- and at most one L-shape, and (c) no two shapes share an edge. By the construction of $H(\Delta_i^{\max})$ every vertex is incident to edges with different angles. In other words, there are no degree-2 vertices whose incident edges both form the same angle with the x-axis. Hence, every vertex lies at the intersection of two shapes or at the bend of a single shape. By Property (b) and the form of the shapes, any two shapes intersect at most once. Thus, it follows from the properties that there are $\mathcal{O}(i^2)$ vertices in total. Moreover, $H(\Delta_i^{\max})$ is planar. Hence, there are $\mathcal{O}(i^2)$ edges.

It remains to show how to cover $H(\Delta_i^{\max})$ with $\mathcal{O}(i)$ shapes such that Properties (a) to (c) hold. We denote the number of L- and J-shapes in $H(\Delta_i^{\max})$ by n_i^L and n_i^J.



(a) $H(\Delta_{i-1}^{\max})$. **(b)** $H(\Delta_{i,1}^{\max})$.

(c) $H(\Delta_{i,2}^{\max})$. **(d)** $H(\Delta_{i}^{\max})$.

Figure 6.10: The shapes covering the edges of the representation of the diagrams. Different shapes are drawn in different colors. The filled disks represent the upper endpoints of the shapes.

We cover $H(\Delta_0^{\max})$ with two J-shapes, one that contains the diagonal edge and one that only contains the vertex with minimum x-coordinate. It is easy to verify that this cover has the desired properties. We have $n_0^L = 0$ and $n_0^J = 2$.

To get from $H(\Delta_{i-1}^{\max})$ to $H(\Delta_i^{\max})$ we perform three steps. Handling the capacity of the edge $v_{i-1}v_i$ may add two L-shapes that consist of the edges on the lines at which we cut the diagrams. In Figure 6.10(b) these are the shapes L_2 and L_3. If some edges were part of a shape S before, which is necessarily an L-shape as the new edges are horizontal, then we remove all horizontal edges from S as they are now covered by the new L-shapes. Thus, no edge is covered twice (Property (c)). This cutting may also shorten or even completely remove some shapes. For example, J_3 is shortened at both ends in Figure 6.10(b). But all vertices on the upper boundary are still the upper endpoints of possibly shortened shapes (Property (a)). In particular, the L-shape

that was part of the upper boundary may be replaced by one of the new L-shapes. Nevertheless, the edges of the upper boundary are still covered by one J- and one L-shape (Property (b)).

Similarly, handling the susceptance may add another L-shape consisting of the horizontal edges on the line $y = 0$. In Figure 6.10(c) the new shape is L_4. This addition may introduce a new vertex on the upper boundary, which is the upper endpoint of the new L-shape. Other than that the upper boundary remains unchanged. Thus, Properties (a) and (b) are maintained. Property (c) can be achieved as in the previous step by shortening existing L-shapes if necessary.

Handling the demand we add a copy of the old upper boundary. By Property (b) this copy can be covered by two shapes (J_4 and L_5 in Figure 6.10(d)). Hence, the new boundary can be covered by duplicates of these. We further add multiple vertical edges, each incident to a vertex on the former upper boundary. Since each such vertex is the upper endpoint of a shape by Property (a), we can simply append the new vertical edges to these shapes without creating any new shapes. Moreover, this ensures that Property (a) holds for the resulting shapes as well.

In total, we add at most four L-shapes and at most one J-shape. Together with $n_0^L = 0$ and $n_0^J = 2$ this implies $n_i^L \leq 4i$ and $n_i^J \leq i + 2$. □

Repeating the three steps for all vertices and edges in the order in which they appear on the cycle, we obtain $H(\Delta_n^{\min})$ and $H(\Delta_n^{\max})$. Note that in A_n the start and end vertex of the path P_n are treated as two different vertices. However, they are actually the same vertex v_0. This implies that in any FACTS flow along P_n the angle difference between them must be 0. Therefore, we are only interested in those values (x, y) such that $(x, y, 0) \in A_n$.

Lemma 6.19. *Let D_{equiv} be the equivalent demand of C with port v_0. Then, $d \in D_{\text{equiv}}$ if and only if there is $(x, y, 0) \in A_n$ with $x - y = d$.*

Proof. By the definition of A_n, any point $(x, y, 0) \in A_n$ corresponds to a FACTS flow in C with $x - y$ units of flow entering C at v. Hence, $x - y \in D_{\text{equiv}}$.

Conversely, any $d \in D_{\text{equiv}}$ corresponds to a FACTS flow f in C where d units enter C at v. These d units split into x units towards v_1 and $-y$ units towards v_{n-1}. Since $v_0 = v_n$, the angle difference between v_0 and v_n is 0 in f. Thus, f witnesses $(x, y, 0) \in A_n$. □

We have $(x, y, 0) \in A_n$ if and only if $\Delta_n^{\min}(x, y) \leq 0 \leq \Delta_n^{\max}(x, y)$ by Lemma 6.15. Let $P_0(A_n)$ be the set of all points (x, y) such that $(x, y, 0) \in A_n$.

Lemma 6.20. *The set $P_0(A_n)$ is a simple polygon, and it can be computed in $\mathcal{O}(n^2)$ time.*

Proof. We compute $P_0(A_n)$ by computing the sets $P_+(A_n) = \{(x, y) \mid \Delta_n^{\max}(x, y) \geq 0\}$ and $P_-(A_n) = \{(x, y) \mid \Delta_n^{\min}(x, y) \leq 0\}$ separately. For brevity, we only describe the procedure for $P_+(A_n)$.

Consider a face h of $H(\Delta_n^{\max})$. There are three cases: the associated values of all incident vertices are all non-negative, they are all negative, or some are negative and some are not. In the first case h is completely included in $P_+(A_n)$. In the second case no point of h belongs to $P_+(A_n)$. And in the third case part of h is included in $P_+(A_n)$. Since all faces are convex and $\Delta_n^{\max}$ is affine on h, there is a line segment that splits h into the part that belongs to $P_+(A_n)$ and the part that does not. We add this line segment as an edge to the graph $H(\Delta_n^{\max})$ (adding vertices for the endpoints if necessary). In all three cases we further mark each edge e of h for which the area next to e in h belongs to $P_+(A_n)$.

Repeating this process for all faces takes $\mathcal{O}(n^2)$ time and yields a graph H'. Some edges of H' are marked only from one side, and these are precisely the edges that bound $P_+(A_n)$. By the monotonicity (Lemma 6.16) and the continuity (Lemma 6.17) of $\Delta_n^{\max}$ they form a cycle, which can be traced in $\mathcal{O}(n^2)$ time.

Similarly, we obtain $P_-(A_n)$ from $H(\Delta_n^{\min})$. Finally, $P_0(A_n)$ is the intersection of $P_+(A_n)$ and $P_-(A_n)$, which can be computed in $\mathcal{O}(n^2)$ time. By the monotonicity of $\Delta_n^{\min}$ and $\Delta_n^{\max}$ and the definitions of $P_-(A_n)$ and $P_+(A_n)$ the intersection is connected. As both $P_+(A_n)$ and $P_-(A_n)$ are simple polygons, $P_0(A_n)$ is a simple polygon as well. □

Each point $(x, y) \in P_0(A_n)$ corresponds to an inflow of $x - y$ into the cycle at v_0. Hence, the interval containing all these values can be determined by computing the minimum and maximum of this difference for all points of $P_0(A_n)$. Since $P_0(A_n)$ is a simple polygon those extrema occur at vertices on the boundary, and it suffices to check these points. This allows us to finally prove Lemma 6.12, which we restate here.

Lemma 6.12. *The equivalent demand D_{equiv} of a cycle C with port v is an interval, and it can be computed in $\mathcal{O}(|C|^3)$ time. Moreover, given some value $d \in D_{\text{equiv}}$, a FACTS flow f in C with $c_f(v) = -d$ can be computed in $\mathcal{O}(|C|^3)$ time.*

Proof. Let n be the number of vertices of C. All steps that operate on $H(\Delta_i^{\min})$ or $H(\Delta_i^{\max})$ take time linear in the size of these graphs. By Lemma 6.18 these graphs have $\mathcal{O}(i^2)$ vertices and edges. To obtain $H(\Delta_n^{\min})$ and $H(\Delta_n^{\max})$ we perform $\mathcal{O}(n)$ steps, which take $\mathcal{O}(n^3)$ time in total. Using the procedure described above, the boundary of the polygons $P_-(A_n)$ and $P_+(A_n)$ can be computed from $H(\Delta_n^{\min})$ and $H(\Delta_n^{\max})$ in time linear in the size of the two graphs. Hence, it takes $\mathcal{O}(n^2)$ time. Moreover, each of the two polygons has $\mathcal{O}(n^2)$ sides. By intersecting the two polygons, we obtain $P_0(A_n)$ in $\mathcal{O}(n^2)$ time. By Lemma 6.19 we have $d \in D_{\text{equiv}}$ if and only if there

is a point $(x, y) \in P_0(A_n)$ such that $x - y = d$. As $P_0(A_n)$ is connected, the equivalent demand is an interval. Thus, it suffices to find the points of $P_0(A_n)$ that minimize and maximize the coordinate difference. Since $P_0(A_n)$ is a polygon and d defined by $d(x, y) = x - y$ is linear, the extrema of d on $P_0(A_n)$ occur at corners of $P_0(A_n)$. Hence, the final step of computing the equivalent demand from $P_0(A_n)$ takes $\mathcal{O}(n^2)$ time. All together, the computation of $H(\Delta_n^{\min})$ and $H(\Delta_n^{\max})$ dominates the running time, which leads to $\mathcal{O}(n^3)$ time in total.

A FACTS flow with a consumption of $-d$ at the port v can be computed by first picking some point $(x, y) \in P_0(A_n)$ with $x - y = -d$, and then retracing how this point was obtained, thereby iteratively computing the flow values. This can be done in $\mathcal{O}(n^3)$ time as well. □

6.4 Conclusion

We studied the FACTS Flow problem, an extension of electrical flows, where the susceptances of the edges may be varied within a given interval. We proved that this problem is $\mathcal{NP}$-complete even for some restricted cases. Previously, hardness results were only known for Maximum FACTS Flow, where additionally the total flow value is to be maximized. Further, we gave an $\mathcal{O}(n^2)$-time algorithm for FACTS Flow with fixed demands in partial 2-trees and an $\mathcal{O}(n^3)$-time algorithm for FACTS Flow with adjustable demands in cacti.

Partial 2-trees are precisely the graphs with treewidth at most 2. Thus, one could investigate whether our result can be extended to graphs of larger treewidth. Another open question is to determine whether FACTS Flow becomes easier if the number of edges with non-constant susceptance intervals is limited. Finally, one may generalize the flows such that the flow on an edge is not directly proportional to the angle difference, but to some function of the angle difference. Such flow formulations are used, e.g., to describe the flow of gas or water [Gro+19].

7 Conclusion

In this thesis we studied problems related to the design and expansion of electrical networks. As described in Chapter 1 the expansion of the electric transmission grid is an important puzzle piece in the energy transition.

Summary. We formalized basic expansion planning problems as Flow Expansion and Electrical Flow Expansion, which differ in the flow model that is used (graph-theoretical vs. electrical flow). In Chapter 3 we analyzed their computational complexities under various restrictions. In the general case both problems are $\mathcal{NP}$-hard. One interesting difference between the two problems is that for Flow Expansion it can be determined in polynomial time whether there is a (not necessarily optimal) expansion, whereas for Electrical Flow Expansion this question is $\mathcal{NP}$-hard to decide. We further presented polynomial-time algorithms for cases in which the network is restricted to certain graph classes. All together, this chapter gives insights into the effect of important network properties on the computational complexities of the two expansion planning problems.

The basic problems do not require the resulting networks to be reliable, but in practical applications this is an important requirement. Therefore, we introduced a reliability criterion based on a criticality measure that takes the dynamic behavior of electrical networks into account (Chapter 4). We formulated this criterion as a set of linear inequalities, which can be added to any expansion planning model. As an example, we defined two expansion planning problems, in which the reliability criterion is added to Electrical Flow Expansion. We investigated the impact this addition has on the solution process and the resulting expansions. Further, we

proposed a greedy heuristic for one of the two problems. This heuristic does not guarantee optimal results, but our experiments indicate that it produces good results faster than an MILP solver.

Electrical Flow Expansion requires candidates to be given as edges of a graph. In particular when designing a new network, such as a microgrid in a remote location, it is not clear which candidates to choose. In Chapter 5 we consider the Microgrid Cable Layout problem, a network design problem where the positions of the generators and consumers are given, but all other vertices and all edges may be freely placed in the plane. We established that the problem is $\mathcal{NP}$-hard, presented a hybrid genetic algorithm, and evaluated it on a set of benchmark instances. The evaluation compares it to heuristics for related planning problems and shows that the hybrid genetic algorithm produces better results.

Instead of expanding the electrical network one may choose to build power electronics with which the electrical flow can be controlled more precisely and the network can be utilized better. This lead us to the notion of FACTS flows. In Chapter 6 we study the complexity of computing a FACTS flow in a network under various restrictions. It is in general $\mathcal{NP}$-complete to decide whether a network admits a FACTS flow. For two restricted cases we presented polynomial-time algorithms.

Conclusion. In Chapter 1 we laid out two goals of this thesis. First, obtain a deeper theoretical understanding of problems related to the design and expansion of electrical flow networks. Second, present algorithmic approaches for such problems. Every chapter contains an analysis of the complexity of the problems studied therein. In particular, Chapters 3 and 6 contain a detailed analysis for problems related to designing and expanding transmission network, such as Electrical Flow Expansion and FACTS Flow. While exploring the complexity of these problems, we also gave exact polynomial-time algorithm for some restricted variants. Moreover, we presented heuristic algorithms for certain network design problems, in particular for the Microgrid Cable Layout problem. Our experimental evaluations show that the algorithms perform well on realistic instances. All in all, we argue that this thesis achieves the goals laid out in Chapter 1.

Outlook. Every chapter contains future work sections related to the content of the chapter. Here, we take a more abstract view on the topics of this thesis and mention possible research topics. In their general forms, the problems we studied in this thesis are $\mathcal{NP}$-hard. From a theoretical perspective it would be interesting to analyze whether approximate solutions to these problems can be computed efficiently. Moreover, in cases where heuristic algorithms are able to compute good solutions fast, it could be interesting to investigate which properties of the instances make this

possible. This could be done both with a theoretical analysis and an experimental evaluation.

In a broader context, we mention in Chapter 1 two other approaches besides grid expansion to deal with increased renewable generation: storage and demand-side management. While there are works that combine grid expansion and storage, a thorough theoretical understanding seems to be lacking. One may expect the combined problem to become more complex. However, sometimes having more options may help when designing efficient algorithms.

Bibliography

[AAF00] Hugo Ambriz-Perez, Enrique Acha, and Claudio R. Fuerte-Esquivel. **Advanced SVC models for Newton-Raphson load flow and Newton optimal power flow studies**. In *IEEE Transactions on Power Systems* volume 15:1, pages 129–136, 2000. DOI: 10.1109/59.852111.
Cited on pages 102, 103.

[AE02] Tawfiq Al-Saba and Ibrahim El-Amin. **The application of artificial intelligent tools to the transmission expansion problem**. In *Electric Power Systems Research* volume 62:2, pages 117–126, 2002. ISSN: 0378-7796. DOI: 10.1016/S0378-7796(02)00037-8.
Cited on page 6.

[Al-21] Fahad S. Al-Ismail. **DC Microgrid Planning, Operation, and Control: A Comprehensive Review**. In *IEEE Access* volume 9, pages 36154–36172, 2021. DOI: 10.1109/ACCESS.2021.3062840.
Cited on page 67.

[AMC03] Natalia Alguacil, Alexis L. Motto, and Antonio J. Conejo. **Transmission expansion planning: a mixed-integer LP approach**. In *IEEE Transactions on Power Systems* volume 18:3, pages 1070–1077, 2003. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2003.814891.
Cited on page 6.

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. **Network Flows: Theory, Algorithms, and Applications**. Upper Saddle River, NJ: Prentice Hall, 1993, Upper Saddle River, NJ. ISBN: 0-13-617549-X.
Cited on pages 11, 12.

[AP03] Abdel-Moamen M. Abdel-Rahim and Narayana P. Padhy. **Power flow control and transmission loss minimization model with TCSC for practical power networks**. In *2003 IEEE Power Engineering Society General Meeting (IEEE Cat. No.03CH37491)*. Volume 2 of, 880–884 Vol. 2, 2003. DOI: 10.1109/PES.2003.1270424.
Cited on pages 102, 103.

[ARK11] Tohid Akbari, Ashkan Rahimikian, and Ahad Kazemi. **A multi-stage stochastic transmission expansion planning method**. In *Energy Conversion and Management* volume 52:8, pages 2844–2853, 2011. ISSN: 0196-8904. DOI: 10.1016/j.enconman.2011.02.023.
Cited on page 45.

[ASSR21] Fernando A. Assis, Iamberg S. Silva, Armando M. Leite da Silva, and Leonidas C. Resende. **Transmission planning with security criteria via enhanced genetic algorithm**. In *Electrical Engineering*, 2021. ISSN: 1432-0487. DOI: 10.1007/s00202-020-01208-y.
Cited on page 6.

[ATCM12] José Aguado, Sebastián de la Torre, Javier Contreras, and Álvaro Martínez. **Planning Long-Term Network Expansion in Electric Energy Systems in Multi-area Settings**. In *Handbook of Networks in Power Systems I, Energy Systems*. 2012, pages 367–393. DOI: 10.1007/978-3-642-23193-3_15.
Cited on pages 45, 78.

[Aus22] Ausgrid. **Demand response Air Conditioning Programs Final Report**. In URL: https://cdn.ausgrid.com.au/-/media/Documents/Demand-Mgmt/DMIA-research/Demand-Response-Airconditioning-Programs-Report-2022.pdf. Tech. rep., 2022.
Cited on page 4.

[BA01] Hans L. Bodlaender and Babette van Antwerpen - de Fluiter. **Parallel Algorithms for Series Parallel Graphs and Graphs with Treewidth Two**. In *Algorithmica* volume 29:4, pages 534–559, 2001. ISSN: 1432-0541. DOI: 10.1007/s004530010070.
Cited on page 21.

[Bar03] Jorge Barreiros. **An hierarchic genetic algorithm for computing (near) optimal Euclidean Steiner trees**. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 56–65. Springer, 2003. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.319.3516&rep=rep1&type=pdf (visited on 2022-01-19).
Cited on pages 68, 78, 80.

[BBC11] Dimitris Bertsimas, David B. Brown, and Constantine Caramanis. **Theory and Applications of Robust Optimization**. In *SIAM Review* volume 53:3, pages 464–501, 2011. DOI: 10.1137/080734510.
Cited on page 6.

[BC19] Paul Bertheau and Catherina Cader. **Electricity sector planning for the Philippine islands: Considering centralized and decentralized supply options**. In *Applied Energy* volume 251, page 113393, 2019. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2019.113393.
Cited on page 68.

[BCB19] Philipp Blechinger, Catherina Cader, and Paul Bertheau. **Least-Cost Electrification Modeling and Planning–A Case Study for Five Nigerian Federal States**. In *Proceedings of the IEEE* volume 107:9, pages 1923–1940, 2019. DOI: 10.1109/JPROC.2019.2924644.
Cited on page 68.

[BCKO08] Marc de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. **Computational Geometry: Algorithms and Applications**. 3rd ed. Springer Berlin, Heidelberg, 2008. ISBN: 978-3-540-77974-2. DOI: 10.1007/978-3-540-77974-2.
Cited on pages 78, 79.

[BGTZ14] Marcus Brazil, Ronald L. Graham, Doreen A. Thomas, and Martin Zachariasen. **On the history of the Euclidean Steiner tree problem**. In *Archive for history of exact sciences* volume 68:3, pages 327–354, Springer, 2014. DOI: 10.1109/TPWRS.2012.2224676.
Cited on pages 68, 73.

[BHS18] Tom Brown, Jonas Hörsch, and David Schlachtberger. **PyPSA: Python for Power System Analysis**. In *Journal of Open Research Software* volume 6:1, 2018. DOI: 10.5334/jors.188.
Cited on pages 4, 6, 55.

[BLS99] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. **Graph Classes: A Survey**. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999, Philadelphia, PA. ISBN: 0-89871-432-X. DOI: 10.1137/1.9780898719796.
Cited on pages 20, 21, 118.

[BOB21] Thomas Bauer, Christian Odenthal, and Alexander Bonk. **Molten Salt Storage for Power Generation**. In *Chemie Ingenieur Technik* volume 93:4, pages 534–546, 2021. DOI: 10.1002/cite.202000137.
Cited on page 3.

[BOGR11] Gerardo Blanco, Fernando Olsina, Francisco Garces, and Christian Rehtanz. **Real Option Valuation of FACTS Investments Based on the Least Square Monte Carlo Method**. In *IEEE Transactions on Power Systems* volume 26:3, pages 1389–1398, 2011. DOI: 10.1109/TPWRS.2010.2094211.
Cited on pages 6, 103.

[Bol98] Béla Bollobás. **Modern Graph Theory**. New York: Springer, 1998, New York. ISBN: 978-0-387-98488-9. DOI: 10.1007/978-1-4612-0619-4.
Cited on pages 16, 17, 54.

[BPG01] Silvio Binato, Mário Veiga F. Pereira, and Sérgio Granville. **A new Benders decomposition approach to solve power transmission network design problems**. In *IEEE Transactions on Power Systems* volume 16:2, pages 235–240, 2001. ISSN: 0885-8950. DOI: 10.1109/59.918292.
Cited on page 45.

[Bra+21a] Jan van den Brand, Yu Gao, Arun Jambulapati, Yin Tat Lee, Yang P. Liu, Richard Peng, and Aaron Sidford. **Faster Maxflow via Improved Dynamic Spectral Vertex Sparsifiers**. 2021. DOI: 10.48550/ARXIV.2112.00722.
Cited on page 15.

[Bra+21b] Jan van den Brand, Yin Tat Lee, Yang P. Liu, Thatchaphol Saranurak, Aaron Sidford, Zhao Song, and Di Wang. **Minimum Cost Flows, MDPs, and ℓ_1-Regression in Nearly Linear Time for Dense Instances**. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC 2021)*, pages 859–869. Association for Computing Machinery, 2021. ISBN: 978-1-4503-8053-9. DOI: 10.1145/3406325.3451108.
Cited on page 15.

[Bra+22] Jan van den Brand, Yu Gao, Arun Jambulapati, Yin Tat Lee, Yang P. Liu, Richard Peng, and Aaron Sidford. **Faster Maxflow via Improved Dynamic Spectral Vertex Sparsifiers**. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2022, pages 543–556. New York, NY, USA: Association for Computing Machinery, 2022. ISBN: 978-1-4503-9264-8. DOI: 10.1145/3519935.3520068.
Cited on page 15.

[Bro+18] Tom Brown, David Schlachtberger, Alexander Kies, Stefan Schramm, and Martin O. W. Greiner. **Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system**. In *Energy* volume 160, pages 720–739, 2018. ISSN: 0360-5442. DOI: 10.1016/j.energy.2018.06.222.
Cited on page 4.

[Bun22] Bundesnetzagentur. **Monitoring des Stromnetzausbaus: Erstes Quartal 2022**. In URL: https://data.netzausbau.de/Vorhaben/Monitoring/Monitoringbericht_Q1-22.pdf. Tech. rep., 2022.
Cited on page 3.

[CC08] Miroslav Chlebík and Janka Chlebíková. **The Steiner tree problem on graphs: Inapproximability results**. In *Theoretical Computer Science* volume 406:3, pages 207–214, 2008. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2008.06.046.
Cited on page 31.

[CCEM20] Silvia Corigliano, Tommaso Carnovali, Darlain Edeme, and Marco Merlo. **Holistic geospatial data-based procedure for electric network design and least-cost energy strategy**. In *Energy for Sustainable Development* volume 58, pages 1–15, 2020. ISSN: 0973-0826. DOI: 10.1016/j.esd.2020.06.008.
Cited on pages 68, 93.

[CH67] Gary Chartrand and Frank Harary. **Planar Permutation Graphs**. In *Annales de l'I.H.P. Probabilités et statistiques* volume 3:4, pages 433–438, Gauthier-Villars, 1967. URL: http://www.numdam.org/item/AIHPB_1967__3_4_433_0/.
Cited on page 21.

[Che+17] Derek M. L. K. Cheong, Tyrone Fernando, Herbert H. C. Iu, Mark Reynolds, and James R. E. Fletcher. **Review of clustering algorithms for microgrid formation**. In *2017 IEEE Innovative Smart Grid Technologies -- Asia (ISGT-Asia)*, pages 1–6, 2017. DOI: 10.1109/ISGT-Asia.2017.8378350.
Cited on page 68.

[Chr+11] Paul Christiano, Jonathan A. Kelner, Aleksander Mądry, Daniel A. Spielman, and Shang-Hua Teng. **Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs**. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*. STOC '11, pages 273–282. New York, NY, USA: ACM, 2011. ISBN: 978-1-4503-0691-1. DOI: 10.1145/1993636.1993674.
Cited on page 16.

[CL01] Takshing Chung and Yuzeng Li. **A Hybrid GA Approach for OPF with Consideration of FACTS Devices**. In *IEEE Power Engineering Review* volume 21:2, pages 47–50, 2001. DOI: 10.1109/MPER.2001.4311272.
Cited on pages 102, 103.

[CM21] Gianfranco Chicco and Andrea Mazza. **Metaheuristics for Transmission Network Expansion Planning**. In *Transmission Expansion Planning: The Network Challenges of the Energy Transition*. Ed. by Sara Lumbreras, Hamdi Abdi, and Andrés Ramos. 1st ed. Springer International Publishing, 2021, pages 13–38. ISBN: 978-3-030-49428-5. DOI: 10.1007/978-3-030-49428-5_2.
Cited on page 6.

[CMT07] Jaeseok Choi, Timothy D. Mount, and Robert J. Thomas. **Transmission Expansion Planning Using Contingency Criteria**. In *IEEE Transactions on Power Systems* volume 22:4, pages 2249–2261, 2007. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2007.908478.
Cited on pages 42, 45.

[CNAO85] Norishige Chiba, Takao Nishizeki, Shigenobu Abe, and Takao Ozawa. **A linear algorithm for embedding planar graphs using PQ-trees**. In *Journal of Computer and System Sciences* volume 30:1, pages 54–76, 1985. ISSN: 0022-0000. DOI: 10.1016/0022-0000(85)90004-2.
Cited on page 20.

[Coh+14] Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. **Solving SDD Linear Systems in Nearly $m \log^{1/2} n$ Time**. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*. STOC '14, pages 343–352. New York, NY, USA: Association for Computing Machinery, 2014. ISBN: 978-1-4503-2710-7. DOI: 10.1145/2591796.2591833.
Cited on page 17.

[CPHL21] Karl-Kiên Cao, Thomas Pregger, Jannik Haas, and Hendrik Lens. **To Prevent or Promote Grid Expansion? Analyzing the Future Role of Power Transmission in the European Energy System**. In *Frontiers in Energy Research* volume 8, 2021. ISSN: 2296-598X. DOI: 10.3389/fenrg.2020.541495.
Cited on pages 4, 5.

[DA16] Shahab Dehghan and Nima Amjady. **Robust Transmission and Energy Storage Expansion Planning in Wind Farm-Integrated Power Systems Considering Transmission Switching**. In *IEEE Transactions on Sustainable Energy* volume 7:2, pages 765–774, 2016. DOI: 10.1109/TSTE.2015.2497336.
Cited on page 6.

[DEG17] Andrés H. Domínguez, Antonio H. Escobar, and Ramón A. Gallego. **An MILP model for the static transmission expansion planning problem including HVAC/HVDC links, security constraints and power losses with a reduced search space**. In *Electric Power Systems Research* volume 143, pages 611–623, 2017. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2016.10.055.
Cited on pages 6, 42.

[Die17] Reinhard Diestel. **Graph Theory**. 5th edition. New York: Springer, 2017, New York. ISBN: 978-3-662-53622-3.
Cited on pages 9, 15.

[DIg22] DIgSILENT GmbH. **PowerFactory**. 2022. URL: https://www.digsilent.de/en/powerfactory.html (visited on 2022-08-31).
Cited on page 4.

[DS14] Center for Discrete Mathematics and Computer Science. **11th DIMACS Implementation Challenge**. 2014. URL: https://dimacs11.zib.de/downloads.html (visited on 2022-02-04).
Cited on page 87.

[Duf65] Richard J. Duffin. **Topology of Series-Parallel Networks**. In *Journal of Mathematical Analysis and Applications* volume 10:2, pages 303–318, Academic Press, 1965. DOI: 10.1016/0022-247X(65)90125-3.
Cited on page 21.

[DZMR16] Andres H. Dominguez, Antonio H. Escobar Zuluaga, Leonardo H. Macedo, and Rubén Romero. **Transmission network expansion planning considering HVAC/HVDC lines and technical losses**. In *2016 IEEE PES Transmission Distribution Conference and Exposition-Latin America (PES T D-LA)*, pages 1–6, 2016. DOI: 10.1109/TDC-LA.2016.7805606.
Cited on page 6.

[EF82] Horst A. Eiselt and H. von Frajer. **On the budget-restricted max flow problem**. In *Operations-Research-Spektrum* volume 3:4, pages 225–231, 1982. DOI: 10.1007/BF01719791.
Cited on page 24.

[Epp92] David Eppstein. **Parallel recognition of series-parallel graphs**. In *Information and Computation* volume 98:1, pages 41–55, 1992. ISSN: 0890-5401. DOI: 10.1016/0890-5401(92)90041-D.
Cited on page 21.

[EREG16] V. Laura M. Escobar, L. Rubén A. Romero, Z. Antonio H. Escobar, and R. Ramón A. Gallego. **Long term transmission expansion planning considering generation-demand scenarios and HVDC lines**. In *2016 IEEE PES Transmission Distribution Conference and Exposition-Latin America (PES T D-LA)*, pages 1–6, 2016. DOI: 10.1109/TDC-LA.2016.7805617.
Cited on page 6.

[Eur+20] European Commission, Directorate-General for Energy, Christopher Andrey, Paul Barberi, Luiza Lacombe, Luc van Nuffel, Frank Gérard, João Gorenstein Dedecca, Koen Rademaekers, Yacine El Idrissi, and Morgan Crenes. **Study on energy storage : contribution to the security of the electricity supply in Europe**. Publications Office, 2020. DOI: 10.2833/077257.
Cited on page 3.

[Eur19] European Commission. **The European Green Deal**. 2019. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1576150542719&uri=COM%3A2019%3A640%3AFIN (visited on 2022-07-21).
Cited on page 1.

[Eur22] Eurostat. **Sustainable development in the European Union – Monitoring report on progress towards the SDGs in an EU context – 2022 edition**. In URL: https://ec.europa.eu/eurostat/en/web/products-statistical-books/-/ks-09-22-019. Tech. rep., 2022.
Cited on page 1.

[Fle+17] James R. E. Fletcher, Tyrone Fernando, Herbert H. C. Iu, Mark Reynolds, and Shervin Fani. **Economic feasibility of stand-alone power systems for existing distribution networks in rural areas**. In *2017 IEEE InnovativeSmart Grid Technologies -- Asia (ISGT-Asia)*, pages 1–5, 2017. DOI: 10.1109/ISGT-Asia.2017.8378356.
Cited on page 66.

[FOF08] Emily B. Fisher, Richard P. O'Neill, and Michael C. Ferris. **Optimal Transmission Switching**. In *IEEE Transactions on Power Systems* volume 23:3, pages 1346–1355, 2008. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2008.922256.
Cited on pages 51, 64.

[FR16] Stephen Frank and Steffen Rebennack. **An introduction to optimal power flow: Theory, formulation, and examples**. In *IIE Transactions* volume 48:12, pages 1172–1197, Taylor & Francis, 2016. DOI: 10.1080/0740817X.2016.1189626.
Cited on pages 15, 56, 64.

[Gar70] Len L. Garver. **Transmission Network Estimation Using Linear Programming**. In *IEEE Transactions on Power Apparatus and Systems* volume PAS-89:7, pages 1688–1697, 1970. ISSN: 0018-9510. DOI: 10.1109/TPAS.1970.292825.
Cited on page 5.

[Gea+21] Juan Gea-Bermúdez, Ida G. Jensen, Marie Münster, Matti Koivisto, Jon G. Kirkerud, Yi-kuang Chen, and Hans Ravn. **The role of sector coupling in the green transition: A least-cost energy system development in Northern-central Europe towards 2050**. In *Applied Energy* volume 289, page 116685, 2021. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2021.116685.
Cited on page 4.

[Gel17] Clark W. Gellings. **Evolving practice of demand-side management**. In *Journal of Modern Power Systems and Clean Energy* volume 5:1, pages 1–9, 2017. DOI: 10.1007/s40565-016-0252-1.
Cited on page 4.

[GG15] Carlos Gamarra and Josep M. Guerrero. **Computational optimization techniques applied to microgrids planning: A review**. In *Renewable and Sustainable Energy Reviews* volume 48, pages 413–424, 2015. ISSN: 1364-0321. DOI: 10.1016/j.rser.2015.04.025.
Cited on page 67.

[GGJ77] Michael R. Garey, Ronald L. Graham, and David S. Johnson. **The Complexity of Computing Steiner Minimal Trees**. In *SIAM Journal on Applied Mathematics* volume 32:4, pages 835–859, 1977. DOI: 10.1137/0132072.
Cited on page 73.

[Gha+21] Ali A. Ghadimi, Mohammad Amani, Mohammad Bayat, Saeid Ahmadi, Mohammad R. Miveh, and Francisco Jurado. **Stochastic transmission expansion planning in the presence of wind farms considering reliability and $N-1$ contingency using grey wolf optimization technique**. In *Electrical Engineering*, 2021. ISSN: 1432-0487. DOI: 10.1007/s00202-021-01339-w. URL: https://doi.org/10.1007/s00202-021-01339-w.
Cited on page 6.

[GJ79] Michael R. Garey and David S. Johnson. **Computers and Intractability, A Guide to the Theory of NP-Completeness**. San Francisco: Freeman, 1979, San Francisco. ISBN: 0-7167-1044-7.
Cited on pages 74, 77.

[GMAD21] Vladimir Z. Gjorgievski, Natasa Markovska, Alajdin Abazi, and Neven Duić. **The potential of power-to-heat demand response to improve the flexibility of the energy system: An empirical review**. In *Renewable and Sustainable Energy Reviews* volume 138, page 110489, 2021. ISSN: 1364-0321. DOI: 10.1016/j.rser.2020.110489.
Cited on page 4.

[Gom+19] Tomas Gomez, Ignacio Herrero, Pablo Rodilla, Rodrigo Escobar, Salvatore Lanza, Ignacio de la Fuente, Maria Luisa Llorens, and Paula Junco. **European Union Electricity Markets: Current Practice and Future View**. In *IEEE Power and Energy Magazine* volume 17:1, pages 20–31, 2019. ISSN: 1558-4216. DOI: 10.1109/MPE.2018.2871739.
Cited on page 56.

[Gra+18] Alban Grastien, Ignaz Rutter, Dorothea Wagner, Franziska Wegner, and Matthias Wolf. **The Maximum Transmission Switching Flow Problem**. In *Proceedings of the 9th ACM e-Energy International Conference on Future Energy Systems (ACM e-Energy'18)*, pages 340–360. ACM Press, 2018. DOI: 10.1145/3208903.3208910.
Cited on pages 24, 26, 37.

[Gro+19] Martin Gross, Marc E. Pfetsch, Lars Schewe, Martin Schmidt, and Martin Skutella. **Algorithmic results for potential-based flows: Easy and hard cases**. In *Networks* volume 73:3, pages 306–324, 2019. DOI: 10.1002/net.21865.
Cited on page 133.

[Gur22] Gurobi Optimization, LLC. **Gurobi Optimizer Reference Manual**. 2022. URL: https://www.gurobi.com.
Cited on pages 56, 77, 88, 93.

[GW22] Max Göttlicher and Matthias Wolf. **A Genetic Algorithm for Finding Microgrid Cable Layouts**. In *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (ACM e-Energy '22)*, pages 1–16. ACM Press, 2022. DOI: 10.1145/3538637.3538835.
Cited on pages 7, 65.

[GYG19] Nnachi Gideon Ude, Hamam Yskandar, and Richards C. Graham. **A Comprehensive State-of-the-Art Survey on the Transmission Network Expansion Planning Optimization Algorithms**. In *IEEE Access*, pages 123158–123181, 2019. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2936682.
Cited on pages 6, 23.

[He91] Xin He. **Efficient parallel algorithms for series parallel graphs**. In *Journal of Algorithms* volume 12:3, pages 409–430, 1991. ISSN: 0196-6774. DOI: 10.1016/0196-6774(91)90012-N.
Cited on page 21.

[Hem21] Reza Hemmati. **Energy Storage Systems in Transmission Expansion Planning**. In *Transmission Expansion Planning: The Network Challenges of the Energy Transition*. Ed. by Sara Lumbreras, Hamdi Abdi, and Andrés Ramos. 1st ed. Cham: Springer International Publishing, 2021, pages 249–268. ISBN: 978-3-030-49428-5. DOI: 10.1007/978-3-030-49428-5_10.
Cited on page 6.

[Hey+19] Evelyn Heylen, Marten Ovaere, Stef Proost, Geert Deconinck, and Dirk Van Hertem. **A multi-dimensional analysis of reliability criteria: From deterministic $N-1$ to a probabilistic approach**. In *Electric Power Systems Research* volume 167, pages 290–300, 2019. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2018.11.001.
Cited on page 43.

[HH14] Chao-Ming Huang and Yann-Chang Huang. **Hybrid optimisation method for optimal power flow using flexible AC transmission system devices**. In *IET Generation, Transmission & Distribution* volume 8:12, pages 2036–2045, 2014. DOI: 10.1049/iet-gtd.2014.0096.
Cited on page 18.

[Hin93] Narain G. Hingorani. **Flexible AC transmission**. In *IEEE Spectrum* volume 30:4, pages 40–45, 1993. ISSN: 0018-9235. DOI: 10.1109/6.206621.
Cited on pages 18, 101.

[HJPW21] Monika Henzinger, Billy Jin, Richard Peng, and David P. Williamson. **Cut-Toggling and Cycle-Toggling for Electrical Flow and Other p-Norm Flows**. 2021. DOI: 10.48550/ARXIV.2109.00653.
Cited on page 18.

[HJW20] Monika Henzinger, Billy Jin, and David P. Williamson. **A Combinatorial Cut-Based Algorithm for Solving Laplacian Linear Systems**. 2020. DOI: 10.48550/ARXIV.2010.16316.
Cited on page 18.

[HLMW16] Daniel Hoske, Dimitar Lukarski, Henning Meyerhenke, and Michael Wegner. **Engineering a Combinatorial Laplacian Solver: Lessons Learned**. In *Algorithms* volume 9:4, 2016. ISSN: 1999-4893. DOI: 10.3390/a9040072.
Cited on page 18.

[HOM22] HOMER Software. **HOMER – Hybrid Renewable and Distributed Generation System Design Software**. 2022. URL: https://www.homerenergy.com/ (visited on 2022-08-31).
Cited on pages 4, 68.

[HT73] John Hopcroft and Robert Tarjan. **Algorithm 447: Efficient Algorithms for Graph Manipulation**. In *Communications of the ACM* volume 16:6, pages 372–378, 1973. ISSN: 0001-0782. DOI: 10.1145/362248.362272.
Cited on pages 22, 32, 33.

[HU16] Andrew H. Hubble and Taha S. Ustun. **Feasibility of microgrid optimization and grid extension for rural electrification**. In *2016 IEEE Region 10 Conference (TENCON)*, pages 1266–1269, 2016. DOI: 10.1109/TENCON.2016.7848215.
Cited on page 66.

[Int21a] International Energy Agency (IAE). **Renewables 2021 – Analysis and forecast to 2026**. In URL: https://www.iea.org/reports/renewables-2021. Tech. rep., 2021.
Cited on page 5.

[Int21b] International Energy Agency (IAE). **World Energy Outlook 2021**. In URL: https://www.iea.org/reports/world-energy-outlook-2021. Tech. rep., Paris, 2021.
Cited on page 2.

[Inv22] Invenergy Transmission LLC. **Grain Belt Express**. 2022. URL: https://grainbeltexpress.com/ (visited on 2022-08-01).
Cited on page 4.

[JJM04] Mário Jesus, Sérgio Jesus, and Alberto Márquez. **Steiner Trees Optimization using Genetic Algorithms**. In URL: https://esght.ualg.pt/sites/ualg.pt/files/ise/tr_2004-01.pdf. Tech. rep., 2004.
Cited on pages 68, 78.

[JWWZ18] Daniel Juhl, David M. Warme, Pawel Winter, and Martin Zachariasen. **The GeoSteiner software package for computing Steiner trees in the plane: an updated computational study**. In *Mathematical Programming Computation* volume 10:4, pages 487–532, Springer, 2018. DOI: 10.1007/s12532-018-0135-8.
Cited on page 77.

[KALT08] Mehrdad A. Kamarposhti, Mostafa Alinezhad, Hamid Lesani, and Nemat Talebi. **Comparison of SVC, STATCOM, TCSC, and UPFC controllers for Static Voltage Stability Evaluated by Continuation Power Flow Method**. In *2008 IEEE Canada Electric Power Conference*, pages 1–8, 2008. DOI: 10.1109/EPC.2008.4763387.
Cited on page 101.

[Kat20] Tarun Kathuria. **A Potential Reduction Inspired Algorithm for Exact Max Flow in Almost $\tilde{O}(m^{4/3})$ Time**. 2020. DOI: 10.48550/ARXIV.2009.03260. URL: https://arxiv.org/abs/2009.03260.
Cited on page 15.

[KHCM18] Jonas A. Kersulis, Ian A. Hiskens, Carleton Coffrin, and Daniel K. Molzahn. **Topological Graph Metrics for Detecting Grid Anomalies and Improving Algorithms**. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7, 2018. DOI: 10.23919/PSCC.2018.8442682.
Cited on page 26.

[Kir47] Gustav R. Kirchhoff. **Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird**. In *Annalen der Physik* volume 148:12, pages 497–508, 1847. DOI: 10.1002/andp.18471481202.
Cited on page 16.

[Kir58] Gustav R. Kirchhoff. **On the Solution of the Equations Obtained from the Investigation of the Linear Distribution of Galvanic Currents**. In *IRE Transactions on Circuit Theory* volume 5:1. Translated by J. B. O'Toole, pages 4–7, 1958. DOI: 10.1109/TCT.1958.1086426.
Cited on page 16.

[KJKM09] Charles Kirubi, Arne Jacobson, Daniel M. Kammen, and Andrew Mills. **Community-Based Electric Micro-Grids Can Contribute to Rural Development: Evidence from Kenya**. In *World Development* volume 37:7, pages 1208–1221, 2009. ISSN: 0305-750X. DOI: 10.1016/j.worlddev.2008.11.005.
Cited on page 66.

[KK62] Harold W. Kuhn and Robert E. Kuenne. **An efficient algorithm for the numerical solution of the generalized Weber problem in spatial economics**. In *Journal of Regional Science* volume 4:2, pages 21–33, 1962. DOI: 10.1111/j.1467-9787.1962.tb00902.x.
Cited on page 80.

[KLS22] Tarun Kathuria, Yang P. Liu, and Aaron Sidford. **Unit Capacity Maxflow in Almost $m^{4/3}$ Time**. In *SIAM Journal on Computing* volume 0:0, FOCS20–175–FOCS20–204, 2022. DOI: 10.1137/20M1383525.
Cited on page 15.

[Koc+16] Burak Kocuk, Hyemin Jeon, Santanu S. Dey, Jeff Linderoth, James Luedtke, and Xu A. Sun. **A Cycle-Based Formulation and Valid Inequalities for DC Power Transmission Problems with Switching**. In *Operations Research* volume 64:4, pages 922–938, 2016. DOI: 10.1287/opre.2015.1471.
Cited on pages 16, 24, 25, 28, 29.

[Koh+19] Peter Kohlhepp, Hassan Harb, Henryk Wolisz, Simon Waczowicz, Dirk Müller, and Veit Hagenmeyer. **Large-scale grid integration of residential thermal energy storages as demand-side flexibility resource: A review of international field studies**. In *Renewable and Sustainable Energy Reviews* volume 101, pages 527–547, 2019. ISSN: 1364-0321. DOI: 10.1016/j.rser.2018.09.045.
Cited on page 4.

[KOPS16] Onur Kahveci, Thomas J. Overbye, Nathan H. Putnam, and Ahmet Soylemezoglu. **Optimization Framework for Topology Design Challenges in Tactical Smart Microgrid Planning**. In *2016 IEEE Power and Energy Conference at Illinois (PECI)*, pages 1–7. Urbana, IL, USA: IEEE, 2016. DOI: 10.1109/PECI.2016.7459262.
Cited on pages 67, 93.

[KOSZ13] Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan A. Zhu. **A Simple, Combinatorial Algorithm for Solving SDD Systems in Nearly-Linear Time**. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*. STOC '13, pages 911–920. Association for Computing Machinery, 2013. ISBN: 978-1-4503-2029-0. DOI: 10.1145/2488608.2488724.
Cited on pages 17, 18.

[KPP04] Hans Kellerer, Ulrich Pferschy, and David Pisinger. **Knapsack problems**. 1st ed. Berlin: Springer, 2004, Berlin. ISBN: 978-3-540-24777-7. DOI: 10.1007/978-3-540-24777-7.
Cited on pages 37, 38.

[Kra17] Oliver Kramer. **Genetic Algorithms Essentials**. Volume 679 of Studies in Computational Intelligence. Cham: Springer, 2017, Cham. ISBN: 978-3-319-52155-8. DOI: 10.1007/978-3-319-52156-5.
Cited on pages 78, 80.

[Kra20] Johann Kraft. **Untersuchung der Kombination von Solar und Wasserkraft zur Off-Grid-Stromversorgung**. Karlsruhe Institute of Technology, Bachelor's thesis. 2020.
Cited on pages 83, 87.

[KSK10] Amin Khodaei, Mohammad Shahidehpour, and Saeed Kamalinia. **Transmission Switching in Expansion Planning**. In *IEEE Transactions on Power Systems* volume 25:3, pages 1722–1733, 2010. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2009.2039946.
Cited on pages 6, 45.

[LA01] Yunqiang Lu and Ali Abur. **Improving system static security via optimal placement of thyristor controlled series capacitors (TCSC)**. In *2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No. 01CH37194)*, 516–521 vol. 2, 2001. DOI: 10.1109/PESW.2001.916901.
Cited on page 103.

[LAR21] Sara Lumbreras, Hamdi Abdi, and Andrés Ramos (editors). **Transmission Expansion Planning: The Network Challenges of the Energy Transition**. 1st ed. Cham: Springer International Publishing, 2021, Cham. ISBN: 978-3-030-49428-5. DOI: 10.1007/978-3-030-49428-5.
Cited on pages 5, 23, 42.

[LARM21] Sara Lumbreras, Hamdi Abdi, Andrés Ramos, and Mansour Moradi. **Introduction: The Key Role of the Transmission Network**. In *Transmission Expansion Planning: The Network Challenges of the Energy Transition*. Ed. by Sara Lumbreras, Hamdi Abdi, and Andrés Ramos. 1st ed. Cham: Springer International Publishing, 2021, pages 1–12. ISBN: 978-3-030-49428-5. DOI: 10.1007/978-3-030-49428-5_1.
Cited on pages 5, 101.

[LBP15] Karsten Lehmann, Russell Bent, and Feng Pan. **Maximizing electrical power supply using FACTS devices**. 2015. DOI: 10.48550/ARXIV.1507.05541.
Cited on pages 18, 19, 102, 104, 105.

[LBWR21] Bowen Li, Sukanta Basu, Simon J. Watson, and Herman W. J. Russchenberg. **A Brief Climatology of Dunkelflaute Events over and Surrounding the North and Baltic Sea Areas**. In *Energies* volume 14:20, 2021. ISSN: 1996-1073. DOI: 10.3390/en14206508.
Cited on pages 2, 3.

[Lei+15] Thomas Leibfried, Tamara Mchedlidze, Nico Meyer-Hübner, Martin Nöllenburg, Ignaz Rutter, Peter Sanders, Dorothea Wagner, and Franziska Wegner. **Operating Power Grids with Few Flow Control Buses**. In *Proceedings of the ACM Sixth International Conference on Future Energy Systems*. e-Energy, pages 289–294. ACM, 2015. ISBN: 978-1-4503-3609-3. DOI: 10.1145/2768510.2768521.
Cited on page 103.

[LG06] Lin Lin and Mitsuo Gen. **Node-Based Genetic Algorithm for Communication Spanning Tree Problem**. In *IEICE transactions on communications* volume E89-B:4, pages 1091–1098, The Institute of Electronics, Information and Communication Engineers, 2006. ISSN: 1745-1345. DOI: 10.1093/ietcom/e89-b.4.1091.
Cited on page 68.

[LGV14] Karsten Lehmann, Alban Grastien, and Pascal Van Hentenryck. **The Complexity of DC-Switching Problems**. 2014. DOI: 10.48550/ARXIV.1411.4369.
Cited on pages 24, 25, 28, 29.

[LGV15] Karsten Lehmann, Alban Grastien, and Pascal Van Hentenryck. **The Complexity of Switching and FACTS Maximum–Potential–Flow Problems**. 2015. DOI: 10.48550/ARXIV.1507.04820.
Cited on pages 102, 104, 105, 110, 111, 112.

[LH00] Tom W. Lambert and Douglas C. Hittle. **Optimization of autonomous village electrification systems by simulated annealing**. In *Solar Energy* volume 68:1, pages 121–132, 2000. ISSN: 0038-092X. DOI: 10.1016/S0038-092X(99)00040-7.
Cited on pages 67, 93.

[LHS19] Zhe Lin, Zechun Hu, and Yonghua Song. **Distribution Network Expansion Planning Considering $N-1$ Criterion**. In *IEEE Transactions on Power Systems* volume 34:3, pages 2476–2478, 2019. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2019.2896841.
Cited on page 42.

[Li+22] Can Li, Antonio J. Conejo, Peng Liu, Benjamin P. Omell, John D. Siirola, and Ignacio E. Grossmann. **Mixed-integer linear programming models and algorithms for generation and transmission expansion planning of power systems**. In *European Journal of Operational Research* volume 297:3, pages 1071–1082, 2022. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2021.06.024.
Cited on page 6.

[Lia+21] Zipeng Liang, Haoyong Chen, Simin Chen, Yongchao Wang, Cong Zhang, and Chongqing Kang. **Robust transmission expansion planning based on adaptive uncertainty set optimization under high-penetration wind power generation**. In *IEEE Transactions on Power Systems* volume 36, pages 2798–2814, 2021. DOI: 10.1109/TPWRS.2020.3045229.
Cited on page 6.

[Lou18] Henry Louie. **Off-Grid Electrical Systems in Developing Countries**. 1st ed. Cham: Springer, 2018, Cham. ISBN: 978-3-319-91890-7. DOI: 10.1007/978-3-319-91890-7.
Cited on page 70.

[LPC20] Zora Luburić, Hrvoje Pandžić, and Miguel Carrión. **Transmission expansion planning model considering battery energy storage, TCSC and lines using AC OPF**. In *IEEE Access* volume 8, pages 203429–203439, 2020. DOI: 10.1109/ACCESS.2020.3036381.
Cited on pages 6, 102, 103.

[LR16a] Sara Lumbreras and Andrés Ramos. **How to solve the transmission expansion planning problem faster: acceleration techniques applied to Benders' decomposition**. In *IET Generation, Transmission & Distribution* volume 10:10, pages 2351–2359, 2016. DOI: 10.1049/iet-gtd.2015.1075.
Cited on page 6.

[LR16b] Sara Lumbreras and Andrés Ramos. **The new challenges to transmission expansion planning. Survey of recent practice and literature review**. In *Electric Power Systems Research* volume 134, pages 19–29, 2016. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2015.10.013.
Cited on page 5.

[LRS14] Sara Lumbreras, Andrés Ramos, and Pedro Sánchez. **Automatic selection of candidate investments for Transmission Expansion Planning**. In *International Journal of Electrical Power & Energy Systems* volume 59, pages 130–140, 2014. ISSN: 0142-0615. DOI: 10.1016/j.ijepes.2014.02.016.
Cited on page 7.

[LS14] Yin Tat Lee and Aaron Sidford. **Path Finding Methods for Linear Programming: Solving Linear Programs in $\tilde{O}(\sqrt{\text{rank}})$ Iterations and Faster Algorithms for Maximum Flow**. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 424–433, 2014. DOI: 10.1109/FOCS.2014.52.
Cited on pages 12, 17.

[LS19] Yin Tat Lee and Aaron Sidford. **Solving Linear Programs with $\tilde{O}(\sqrt{\text{rank}})$ Linear System Solves**. 2019. DOI: 10.48550/ARXIV.1910.08033.
Cited on pages 12, 17.

[LS20a] Yang P. Liu and Aaron Sidford. **Faster Divergence Maximization for Faster Maximum Flow**. 2020. DOI: 10.48550/ARXIV.2003.08929.
Cited on page 15.

[LS20b] Yang P. Liu and Aaron Sidford. **Faster Energy Maximization for Faster Maximum Flow**. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. New York, NY, USA: Association for Computing Machinery, 2020, pages 803–814. ISBN: 978-1-4503-6979-4. DOI: 10.1145/3357713.3384247.
Cited on page 15.

[Luh+22] Matthias Luh, Kaleb Phipps, Anthony Britto, Matthias Wolf, Marek Lutz, and Johann Kraft. **High-Resolution Real-World Electricity Data from ThreeMicrogrids in the Global South**. In *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems (ACM e-Energy '22)*, pages 496–514. ACM Press, 2022. DOI: 10.1145/3538637.3539763.
Cited on page 97.

[Mąd16] Aleksander Mądry. **Computing Maximum Flow with Augmenting Electrical Flows**. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 593–602, 2016. DOI: 10.1109/FOCS.2016.70.
Cited on page 15.

[MC21] Mahdi Mehrtash and Yankai Cao. **A New Global Solver for Transmission Expansion Planning with AC Network Model**. In *IEEE Transactions on Power Systems*, pages 282–293, 2021. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2021.3086085.
Cited on page 24.

[Mch+15] Tamara Mchedlidze, Martin Nöllenburg, Ignaz Rutter, Dorothea Wagner, and Franziska Wegner. **Towards Realistic Flow Control in Power Grid Operation**. In *Energy Informatics*. Ed. by Sebastian Gottwalt, Lukas König, and Hartmut Schmeck, pages 192–199. Cham: Springer International Publishing, 2015. ISBN: 978-3-319-25876-8. DOI: 10.1007/978-3-319-25876-8_16.
Cited on pages 102, 103.

[Mey+18] Nico Meyer-Huebner, Sebastian Weck, Florian Bennewitz, Michael Suriyah, Kalpesh Bhalodi, Marco Giuntoli, Veronica Biagini, Athanasios Krontiris, Andreas Wasserrab, Mario Ndreko, Michael Wiest, Thomas Leibfried, and Jutta Hanson. **$N-1$-Secure Dispatch Strategies of Embedded HVDC Using Optimal Power Flow**. In *2018 IEEE Power Energy Society General Meeting (PESGM)*, pages 1–5, 2018. DOI: 10.1109/PESGM.2018.8586478.
Cited on page 42.

[MHN22] Samuel Marrero-Vera, Mario Hernandez-Tejera, and Ignacio Nuez-Pestana. **Pareto optimality for FACTS devices placement considering demand variations**. In *Electric Power Systems Research* volume 211, page 108177, 2022. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2022.108177.
Cited on page 103.

[Mit79] Sandra L. Mitchell. **Linear algorithms to recognize outerplanar and maximal outerplanar graphs**. In *Information Processing Letters* volume 9:5, pages 229–232, 1979. ISSN: 0020-0190. DOI: 10.1016/0020-0190(79)90075-9.
Cited on page 21.

[MKMC22] Reza Mahroo, Amin Kargarian, Mahdi Mehrtash, and Antonio J. Conejo. **Robust Dynamic TEP with an $N-c$ Security Criterion: A Computationally Efficient Model**. In *IEEE Transactions on Power Systems*, pages 1–1, 2022. DOI: 10.1109/TPWRS.2022.3163961.
Cited on page 24.

[MM17] Riham Moharam and Ehab Morsy. **Genetic algorithms to balanced tree structures in graphs**. In *Swarm and Evolutionary Computation* volume 32, pages 132–139, 2017. ISSN: 2210-6502. DOI: 10.1016/j.swevo.2016.06.005.
Cited on pages 78, 79, 80.

[MNMR20] Marco Meneses, Edgar Nascimento, Leonardo H. Macedo, and Rubén Romero. **Transmission Network Expansion Planning Considering Line Switching**. In *IEEE Access* volume 8, pages 115148–115158, 2020. DOI: 10.1109/ACCESS.2020.3003973.
Cited on page 6.

[MPS10] Luciano S. Moulin, Michael Poss, and Claudia Sagastizábal. **Transmission expansion planning with re-design**. In *Energy Systems* volume 1:2, pages 113–139, 2010. ISSN: 1868-3975. DOI: 10.1007/s12667-010-0010-9.
Cited on pages 6, 7, 24, 25, 31.

[MR18] Aristide Mingozzi and Roberto Roberti. **An Exact Algorithm for the Fixed Charge Transportation Problem Based on Matching Source and Sink Patterns**. In *Transportation Science* volume 52:2, pages 229–238, 2018. DOI: 10.1287/trsc.2017.0742.
Cited on page 25.

[MSA15] Alexandre Moreira, Alexandre Street, and José M. Arroyo. **An Adjustable Robust Optimization Approach for Contingency-Constrained Transmission Expansion Planning**. In *IEEE Transactions on Power Systems* volume 30:4, pages 2013–2022, 2015. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2014.2349031.
Cited on pages 42, 64.

[MSAR18] Meisam Mahdavi, Carlos Sabillon Antunez, Majid Ajalli, and Rubén Romero. **Transmission Expansion Planning: Literature Review and Classification**. In *IEEE Systems Journal*, pages 1–12, 2018. ISSN: 1932-8184. DOI: 10.1109/JSYST.2018.2871793.
Cited on pages 6, 23.

[Muñ+21] Gregorio Muñoz-Delgado, Javier Contreras, José M. Arroyo, Agustin Sanchez de la Nieta, and Madeleine Gibescu. **Integrated Transmission and Distribution System Expansion Planning under Uncertainty**. In *IEEE Transactions on Smart Grid*, pages 4113–4125, 2021. ISSN: 1949-3061. DOI: 10.1109/TSG.2021.3071385.
Cited on page 6.

[NB19] Fabian Neumann and Tom Brown. **Heuristics for Transmission Expansion Planning in Low-Carbon Energy System Models**. In *2019 16th International Conference on the European Energy Market (EEM)*, pages 1–8, 2019. DOI: 10.1109/EEM.2019.8916411.
Cited on pages 6, 24.

[NHB22] Fabian Neumann, Veit Hagenmeyer, and Tom Brown. **Assessments of linear power flow and transmission loss approximations in coordinated capacity expansion problems**. In *Applied Energy* volume 314, page 118859, 2022. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2022.118859.
Cited on page 5.

[NS 20] NS Energy. **Electric Thermal Energy Storage (ETES) System, Hamburg**. 2020. URL: https://www.nsenergybusiness.com/projects/electric-thermal-energy-storage-etes-system-hamburg/ (visited on 2022-08-29).
Cited on pages 3, 4.

[NSRF17] Steven Nolan, Scott Strachan, Puran Rakhra, and Damien Frame. **Optimized Network Planning of Mini-Grids for the Rural Electrification of Developing Countries**. In *2017 IEEE PES PowerAfrica*, pages 489–494. Accra, Ghana: IEEE, 2017. DOI: 10.1109/PowerAfrica.2017.7991274.
Cited on page 68.

[OR14] David Oertel and Ramamoorthi Ravi. **Complexity of transmission network expansion planning**. In *Energy Systems* volume 5:1, pages 179–207, 2014. ISSN: 1868-3975. DOI: 10.1007/s12667-013-0091-3.
Cited on page 7.

[Orl13] James B. Orlin. **Max Flows in $\mathcal{O}(nm)$ Time, or Better**. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 765–774. ACM, 2013. ISBN: 978-1-4503-2029-0. DOI: 10.1145/2488608.2488705.
Cited on page 34.

[Pac21] PacifiCorp. **Gateway West**. 2021. URL: https://www.pacificorp.com/transmission/transmission-projects/energy-gateway/gateway-west.html (visited on 2022-08-01).
Cited on page 4.

[PRW18] Chitaranjan Phurailatpam, Bharat Singh Rajpurohit, and Lingfeng Wang. **Planning and optimization of autonomous DC microgrids for rural and urban applications in India**. In *Renewable and Sustainable Energy Reviews* volume 82, pages 194–204, 2018. ISSN: 1364-0321. DOI: 10.1016/j.rser.2017.09.022.
Cited on page 68.

[RBM15] Roberto Roberti, Enrico Bartolini, and Aristide Mingozzi. **The Fixed Charge Transportation Problem: An Exact Algorithm Based on a New Integer Programming Formulation**. In *Management Science* volume 61:6, pages 1275–1291, 2015. DOI: 10.1287/mnsc.2014.1947.
Cited on page 25.

[Ref+21] Mohamed M. Refaat, Shady H. E. Abdel Aleem, Yousry Atia, Ziad M. Ali, and Mahmoud M. Sayed. **AC and DC Transmission Line Expansion Planning Using Coronavirus Herd Immunity Optimizer**. In *2021 22nd International Middle East Power Systems Conference (MEPCON)*, pages 313–318, 2021. DOI: 10.1109/MEPCON50283.2021.9686191.
Cited on page 6.

[Ren22a] Renewable Energy Magazine. **First Commercial Sand-based Thermal Energy Storage Is in Operation**. 2022. URL: https://www.renewableenergymagazine.com/storage/first-commercial-sandbased-thermal-energy-storage-is-20220707 (visited on 2022-08-29).
Cited on page 4.

[Ren22b] Renewable Energy Policy Network for the 21st Century (REN21. **Renewables 2022: Global Status Report**. In URL: https://www.ren21.net/reports/global-status-report/. Tech. rep., Paris, 2022.
Cited on page 1.

[RFRW21] Manou Rosenberg, Tim French, Mark Reynolds, and Lyndon While. **A Genetic Algorithm Approach for the Euclidean Steiner Tree Problem with Soft Obstacles**. In *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO '21, pages 618–626. Association for Computing Machinery, 2021. ISBN: 978-1-4503-8350-9. DOI: 10.1145/3449639.3459397.
Cited on page 69.

[RFRW22] Manou Rosenberg, Tim French, Mark Reynolds, and Lyndon While. **Finding an optimised infrastructure for electricity distribution networks in rural areas - A comparison of different approaches**. In *Swarm and Evolutionary Computation* volume 68, page 101018, 2022. ISSN: 2210-6502. DOI: 10.1016/j.swevo.2021.101018.
Cited on pages 66, 68.

[RHS18] Hans-Kristian Ringkjøb, Peter M. Haugan, and Ida M. Solbrekke. **A review of modelling tools for energy and electricity systems with large shares of variable renewables**. In *Renewable and Sustainable Energy Reviews* volume 96, pages 440–459, 2018. ISSN: 1364-0321. DOI: 10.1016/j.rser.2018.08.002.
Cited on page 4.

[SC98] Ge Shaoyun and Takshing Chung. **Optimal active power flow incorporating FACTS devices with power flow control constraints**. In *International Journal of Electrical Power & Energy Systems* volume 20:5, pages 321–326, 1998. ISSN: 0142-0615. DOI: 10.1016/S0142-0615(97)00081-1.
Cited on pages 102, 103.

[Sch98] Alexander Schrijver. **Theory of linear and integer programming**. John Wiley & Sons, 1998. ISBN: 0-471-98232-6.
Cited on page 104.

[Sie22] Siemens Gamesa Renewable Energy. **Industrial Decarbonization using Electric Thermal Energy Storage (ETES)**. 2022. URL: https://www.siemensgamesa.com/products-and-services/hybrid-and-storage/thermal-energy-storage-with-etes-add (visited on 2022-08-01).
Cited on page 3.

[SMB10] Hossein Shayeghi, Meisam Mahdavi, and Amir Bagheri. **Discrete PSO algorithm based optimization of transmission lines loading in TNEP problem**. In *Energy Conversion and Management* volume 51:1, pages 112–121, 2010. ISSN: 0196-8904. DOI: 10.1016/j.enconman.2009.08.030.
Cited on page 42.

[SRG14] John Shortle, Steffen Rebennack, and Fred W. Glover. **Transmission-Capacity Expansion for Minimizing Blackout Probabilities**. In *IEEE Transactions on Power Systems* volume 29:1, pages 43–52, 2014. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2013.2279508.
Cited on page 43.

[SRS21] Sayed Abdullah Sadat, Xinyang Rui, and Mostafa Sahraei-Ardakani. **Computational Impacts of SVCs on Optimal Power Flow using Approximated Active-Set Interior Point Algorithm**. In *2021 North American Power Symposium (NAPS)*, pages 1–6, 2021. DOI: 10.1109/NAPS52732.2021.9654705.
Cited on pages 102, 103.

[SS20] Kavita Singh and Shyam Sundar. **A hybrid genetic algorithm for the degree-constrained minimum spanning tree problem**. In *Soft Computing* volume 24:3, pages 2169–2186, 2020. ISSN: 1433-7479. DOI: 10.1007/s00500-019-04051-x.
Cited on page 68.

[SSH12] Katrin Schaber, Florian Steinke, and Thomas Hamacher. **Transmission grid extensions for the integration of variable renewable energies in Europe: Who benefits where?** In *Energy Policy* volume 43, pages 123–135, 2012. ISSN: 0301-4215. DOI: 10.1016/j.enpol.2011.12.040.
Cited on page 4.

[ST04] Daniel A. Spielman and Shang-Hua Teng. **Nearly-Linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems**. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing (STOC '04)*, pages 81–90. Association for Computing Machinery, 2004. ISBN: 1-58113-852-0. DOI: 10.1145/1007352.1007372.
Cited on page 17.

[ST14] Daniel A. Spielman and Shang-Hua Teng. **Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems**. In *SIAM Journal on Matrix Analysis and Applications* volume 35:3, pages 835–885, 2014. DOI: 10.1137/090771430.
Cited on page 17.

[Sum+14] Jon Sumanik-Leary, Milan Delor, Matt Little, Martin Bellamy, Arthur Williams, and Sam Williamson. **Engineering in Development: Energy**. 2014. URL: https://thewindyboy.files.wordpress.com/2014/10/energy-book-final-for-press.pdf (visited on 2021-12-15).
Cited on pages 66, 70.

[Tec19] Engineers Without Borders – Karlsruhe Institute of Technology. **Hydroélectricité Idjwi: Bericht zur ersten Bauphase, September – Dezember 2018**. In URL: https://www.ludwig-boelkow-stiftung.org/wp-content/uploads/2019/08/EWB-Kongo-Bericht_Bauphase1.pdf. Tech. rep., 2019.
Cited on page 97.

[Thu+18] Leon Thurner, Alexander Scheidler, Florian Schäfer, Jan-Hendrik Menke, Julian Dollichon, Friederike Meier, Steffen Meinecke, and Martin Braun. **Pandapower—An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems**. In *IEEE Transactions on Power Systems* volume 33:6, pages 6510–6521, 2018. DOI: 10.1109/TPWRS.2018.2829021.
Cited on page 4.

[Uni15a] United Nations. **Paris Agreement**. 2015. URL: https://unfccc.int/sites/default/files/english_paris_agreement.pdf (visited on 2022-07-20).
Cited on page 1.

[Uni15b] United Nations General Assembly. **Transforming our world: the 2030 Agenda for Sustainable Development**. A/RES/70/1. 2015. URL: https://sdgs.un.org/2030agenda (visited on 2021-12-06).
Cited on pages 1, 66, 174.

[Uni21] United Nations. **The Sustainable Development Goals Report 2021**. 2021. URL: https://unstats.un.org/sdgs/report/2021/ (visited on 2021-12-14).
Cited on pages 1, 66.

[VBP13] Jonas C. Villumsen, Geir Brønmo, and Andy B. Philpott. **Line capacity expansion and transmission switching in power systems with large-scale wind power**. In *IEEE Transactions on Power Systems* volume 28:2, pages 731–739, 2013. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2012.2224143.
Cited on page 6.

[Vis21] Vistra. **Vistra Completes Expansion of Battery Energy Storage System at its Flagship California Facility**. 2021. URL: https://investor.vistracorp.com/download/Vistra+Moss+Landing+Phase+II+Complete+-+Aug+2021_FINAL.pdf (visited on 2022-08-01).
Cited on page 3.

[VM05] Mallikarjuna R. Vallem and Joydeep Mitra. **Siting and sizing of distributed generation for optimal microgrid architecture**. In *Proceedings of the 37th Annual North American Power Symposium, 2005.* Pages 611–616. Ames, IA, USA: IEEE, 2005. DOI: 10.1109/NAPS.2005.1560597.
Cited on page 68.

[VMP06] Mallikarjuna R. Vallem, Joydeep Mitra, and Shashi B. Patra. **Distributed Generation Placement for Optimal Microgrid Architecture**. In *2005/2006 IEEE/PES Transmission and Distribution Conference and Exhibition*, pages 1191–1195. IEEE, 2006. DOI: 10.1109/TDC.2006.1668674.
Cited on page 68.

[VP12] Jonas C. Villumsen and Andy B. Philpott. **Investment in electricity networks with transmission switching**. In *European Journal of Operational Research* volume 222:2, pages 377–385, 2012. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2012.05.002.
Cited on page 6.

[VSC22] Phillipe Vilaça, Alexandre Street, and J. Manuel Colmenar. **A MILP-based heuristic algorithm for transmission expansion planning problems**. In *Electric Power Systems Research* volume 208, page 107882, 2022. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2022.107882.
Cited on pages 7, 24.

[Wag37] Klaus Wagner. **Über eine Eigenschaft der ebenen Komplexe**. German. In *Mathematische Annalen* volume 114:1, pages 570–590, 1937. ISSN: 1432-1807. DOI: 10.1007/BF01594196.
Cited on page 20.

[WC83] Joseph A. Wald and Charles J. Colbourn. **Steiner Trees, Partial 2-Trees, and Minimum IFI Networks**. In *Networks* volume 13:2, pages 159–167, 1983. DOI: 10.1002/net.3230130202.
Cited on pages 21, 117, 120.

[Win17] Wind Europe. **Wind energy and on-site energy storage: exploring market opportunities**. 2017. URL: https://windeurope.org/wp-content/uploads/files/policy/position-papers/WindEurope-Wind-energy-and-on-site-energy-storage.pdf (visited on 2022-08-01).
Cited on page 3.

[Win21] Wind Europe. **Wind energy in Europe: 2021 Statistics and the outlook for 2022–2026**. In URL: https://windeurope.org/intelligence-platform/product/wind-energy-in-europe-2021-statistics-and-the-outlook-for-2022-2026/. Tech. rep., 2021.
Cited on page 5.

[Wit+16] Dirk Witthaut, Martin Rohden, Xiaozhu Zhang, Sarah Hallerberg, and Marc Timme. **Critical Links and Nonlocal Rerouting in Complex Supply Networks**. In *Physical Review Letters* volume 116:13, page 138701, American Physical Society, 2016. DOI: 10.1103/PhysRevLett.116.138701.
Cited on pages 41, 42, 43, 46, 49, 50, 56, 63, 179.

[Wu+18] Zhi Wu, Xiao Du, Wei Gu, Xiao-Ping Zhang, and Junjie Li. **Automatic Selection Method for Candidate Lines in Transmission Expansion Planning**. In *IEEE Access* volume 6, pages 11605–11613, 2018. DOI: 10.1109/ACCESS.2018.2798063.
Cited on page 7.

[WW20] Dorothea Wagner and Matthias Wolf. **Preventing Critical Edges when Designing Transmission Networks**. In *Proceedings of the Eleventh ACM International Conference on Future Energy Systems (ACM e-Energy '20)*, pages 264–278. ACM Press, 2020. DOI: 10.1145/3396851.3397735.
Cited on pages 7, 41.

[WW21] Dorothea Wagner and Matthias Wolf. **The Complexity of Flow Expansion and Electrical Flow Expansion**. In *Proceedings of the 47th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'21)*. Ed. by Tomáš Bureš, Riccardo Dondi, Johann Gamper, Giovanna Guerrini, Tomasz Jurdzinski, Claus Pahl, Florian Sikora, and Prudence W. H. Wong. Lecture Notes in Computer Science, pages 431–441. Springer, 2021. DOI: 10.1007/978-3-030-67731-2_32.
Cited on pages 7, 23, 25.

[YL20] Mingjia Yin and Kang Li. **Optimal Allocation of Distributed Generations with SOP in Distribution Systems**. In *2020 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5, 2020. DOI: 10.1109/PESGM41954.2020.9281656.
Cited on page 47.

[ZA05] Marek Zima and Göran Andersson. **On Security Criteria in Power Systems Operation**. In *IEEE Power Engineering Society General Meeting, 2005.* Volume 3 of, pages 3089–3093, 2005. DOI: 10.1109/PES.2005.1489533.
Cited on pages 42, 43, 48.

[ZCZ17] Junpeng Zhan, C. Y. Chung, and Alireza Zare. **A Fast Solution Method for Stochastic Transmission Expansion Planning**. In *IEEE Transactions on Power Systems* volume 32:6, pages 4684–4695, 2017. DOI: 10.1109/TPWRS.2017.2665695.
Cited on page 6.

[ZL99] Xiaoxin Zhou and Jun Liang. **Overview of control schemes for TCSC to enhance the stability of power systems**. In *IEE Proceedings - Generation, Transmission and Distribution* volume 146:2, 125–130(5), 1999. ISSN: 1350-2360. DOI: 10.1049/ip-gtd:19990062.
Cited on page 101.

[ZLC19] Junpeng Zhan, Weijia Liu, and Chi Y. Chung. **Stochastic Transmission Expansion Planning Considering Uncertain Dynamic Thermal Rating of Overhead Lines**. In *IEEE Transactions on Power Systems* volume 34:1, pages 432–443, 2019. DOI: 10.1109/TPWRS.2018.2857698.
Cited on pages 6, 24.

[ZMT11] Ray D. Zimmerman, Carlos E. Murillo-Sanchez, and Robert J. Thomas. **MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education**. In *IEEE Transactions on Power Systems* volume 26:1, pages 12–19, 2011. ISSN: 0885-8950. DOI: 10.1109/TPWRS.2010.2051168.
Cited on page 4.

[Zop+22] Reinaldo T. Zoppei, Marcos A. J. Delgado, Leonardo H. Macedo, Marcos J. Rider, and Rubén Romero. **A Branch and Bound Algorithm for Transmission Network Expansion Planning Using Nonconvex Mixed-Integer Nonlinear Programming Models**. In *IEEE Access* volume 10, pages 39875–39888, 2022. DOI: 10.1109/ACCESS.2022.3166153.
Cited on page 24.

List of Figures

List of Tables

List of Acronyms

AC	Alternating current Used on pages 5, 6, 15, 24, 98, 103
CC-TNEP	Criticality-Constrained Transmission Network Expansion Planning Used on pages 43, 53, 54, 57–59, 62–64, 151, 155
CME	Criticality Minimal Expansion Used on pages 43, 53, 54, 59–62, 64, 153, 156
DC	Direct current Used on pages 3, 5, 15, 16, 101
DFS	Depth-first search Used on pages 32, 81, 82
EU	European Union Used on pages 1, 3
FACTS	Flexible AC Transmission System Used on pages iv, 6, 8, 9, 18, 19, 101–114, 121–125, 128, 132–134, 136
IEA	International Energy Agency Used on pages 2, 5

MILP	Mixed-integer linear program Used on pages iv, 6–8, 41–43, 45, 46, 53, 54, 56, 61, 82, 83, 86–88, 93, 95, 102, 104, 105, 136, 151, 153
MINLP	Mixed-integer non-linear program Used on page 7
MP-EFE	Multi-Period Electrical Flow Expansion Used on pages 44–46, 50, 53, 54, 57, 58, 62–64, 155
SDD	Symmetric diagonally dominant Used on page 17
SDG	Sustainable development goal formulated by the United Nations [Uni15b] Used on pages 1, 66
SVC	Static VAR (volt-ampere reactive) compensator Used on page 103
TCSC	Thyristor controlled series capacitator Used on page 103
UN	United Nations Used on pages 1, 66
USA	United States of America Used on pages 3, 4

List of Symbols

Number Sets

$\mathbb{N}$ The set of all natural numbers excluding 0
Used on pages 22, 29–31, 69, 74

$\mathbb{N}_0$ The set of all natural numbers including 0
Used on pages 35–38

$\mathbb{Q}$ The set of all rational numbers
Used on pages 102, 105

$\mathbb{Q}_{\geq 0}$ The set $\{x \in \mathbb{Q} \mid x \geq 0\}$ of all non-negative rational numbers
Used on pages 73, 111

$\mathbb{Q}_{>0}$ The set $\{x \in \mathbb{Q} \mid x > 0\}$ of all positive rational numbers
Used on pages 102, 106, 107

$\mathbb{R}$ The set of all real numbers
Used on pages 10, 11, 15, 16, 19, 44, 51, 69, 73, 74, 102, 114–117, 124–126, 152, 154

$\mathbb{R}_{\geq 0}$ The set $\{x \in \mathbb{R} \mid x \geq 0\}$ of all non-negative real numbers
Used on pages 12, 19, 20, 44, 53, 69, 73, 74, 82, 84

$\mathbb{R}_{>0}$ The set $\{x \in \mathbb{R} \mid x > 0\}$ of all positive real numbers
Used on pages 10, 11, 16, 19, 44, 47, 102

Complexity Classes

$\mathcal{APX}$ The set of optimization problems for which there is a polynomial-time constant-factor approximation algorithm
Used on page 25

$\mathcal{NP}$ The set of decision problems that can be decided by a non-deterministic Turing machine in polynomial time
Used on pages iii, iv, 7, 8, 19, 23–26, 28–31, 35, 37, 39, 45, 65, 67, 73, 74, 77, 101, 102, 104, 105, 111, 113, 122, 134–136

$\mathcal{P}$ The set of decision problems that can be decided by a deterministic Turing machine in polynomial time
Used on page 37

Basic Variables

B The function that maps edges to susceptance intervals
Used on pages 19, 102, 106, 108, 110, 113, 114, 120, 128, 129

b The function that maps edges to susceptances
Used on pages 16–19, 37, 38, 44, 45, 50, 102, 109–111, 120, 125, 129, 151, 153

$b_{\min}$ The function that maps edges to the minimum of the susceptance interval
Used on pages 115, 120

$b_{\max}$ The function that maps edges to the maximum of the susceptance interval
Used on pages 115, 120

C The set of cable types given as input to Microgrid Cable Layout
Used on pages 69, 70, 72, 73, 82, 83

c The function $c\colon E_{\text{cand}} \to \mathbb{R}_{\geq 0}$ that maps the candidate edges to their costs
Used on pages 19, 30, 36–38, 44, 45, 53, 54, 151, 154

c_{equip} The function that maps the degree of a vertex and the maximum power of an edge incident to the vertex to the costs of the equipment at this vertex
Used on pages 72, 73

c_f The function that maps each vertex to its actual consumption in a given flow f
Used on pages 10, 14, 15, 27, 35, 103, 110, 111, 122, 123, 125, 133

c_{line} The function that maps the cable types to their costs per meter
Used on pages 69, 72–75, 83

c_{pole} The cost of a pole given as input to Microgrid Cable Layout
Used on pages 72–74

cap The function that maps the edges to their capacities
Used on pages 10, 11, 14, 15, 19, 29, 30, 36–38, 44–47, 50–52, 55, 69, 71, 73–75, 83, 102, 106, 108, 110, 114, 115, 120, 123, 126–128, 151–153

$\mathrm{cap}^{\mathrm{res}}$ The function that maps the edges of the residual network to their capacities
Used on page 46

cost The total cost of a network or an expansion
Used on pages 19, 30, 31, 44

$\mathrm{Cost}_{\max}$ The budget in Criticality Minimal Expansion
Used on pages 53, 54, 154

crit The function that maps the edges to their criticalities
Used on pages 47, 48, 53, 54

$\mathrm{Crit}_{\max}$ The maximum total criticality gives as input to Criticality-Constrained Transmission Network Expansion Planning
Used on pages 53, 54, 57, 62, 152

D The function that maps the vertices to their demand intervals
Used on pages 10–15, 19, 27, 29, 30, 34, 35, 44, 102, 108, 111, 113, 114, 123, 125, 126, 128, 129

d The function that maps the vertices to their demands
Used on pages 11, 17, 32, 35, 36, 38, 44, 45, 50, 51, 69, 70, 72–75, 81, 82, 84, 85, 105, 106, 108, 114, 120, 129, 151, 153

D_{equiv} The equivalent demand of a subnetwork.
Used on pages 122, 123, 132, 133

$\hat{d}_{\max}$ The maximum of the total negative demand of sources and the total demand of sinks.
Used on pages 13–15

$\hat{d}^S_{\min}$ The sum of the negative of the maximum demands of the sources. This is a lower bound for the total amount of flow produced by the sources in any flow.
Used on pages 13, 14

$\hat{d}^T_{\min}$ The sum of the minimum demands of the sinks. This is a lower bound for the amount of total flow consumed by the sinks in any flow.
Used on pages 13, 14

d_{tot} The sum of all demands in a flow network with fixed demands.
Used on pages 114, 120, 121

$\Delta^{\max}_A$ A boundary function of a reduced diagram A defined by $\Delta^{\max}_A(y) = \max\{\Delta \mid (y, \Delta) \in A\}$
Used on pages 116–118

$\Delta_i^{\max}$ The function representing the upper hull of A_i defined by $\Delta_i^{\max}(x, y) = \max\{\Delta \mid (x, y, \Delta) \in A_i\}$
Used on pages 124–134

$\Delta_A^{\min}$ A boundary function of a reduced diagram A defined by $\Delta_A^{\min}(y) = \min\{\Delta \mid (y, \Delta) \in A\}$
Used on pages 116–118

$\Delta_i^{\min}$ The function representing the lower hull of A_i defined by $\Delta_i^{\min}(x, y) = \min\{\Delta \mid (x, y, \Delta) \in A_i\}$
Used on pages 124–130, 132–134

E The set of edges of a graph
Used on pages 9–11, 13–19, 31, 37, 44, 45, 52–55, 69–76, 79–85, 102, 108, 110, 151–154

$\overline{E}$ The underlying set of undirected edges
Used on pages 9, 10, 16, 69, 82–85

E_{cand} The set of candidate edges in an expansion problem
Used on pages 19, 28, 36, 37, 44, 45, 52, 54, 55, 151–154

E_{ex} The set of existing edges in an expansion problem
Used on pages 19, 28, 36, 44, 45, 52, 54, 55, 151–153

E^{res} The edges of a residual network
Used on page 46

f A flow, which maps the edges to values in $\mathbb{R}$
Used on pages 10–19, 27, 28, 32, 35, 38, 45–53, 55, 102, 103, 106–111, 114, 122–126, 132, 133, 151–154

f^{res} A flow in the residual network
Used on pages 51–53

G A directed graph
Used on pages 9–11, 13, 16, 17, 19, 21, 22, 31–33, 35–37, 44, 45, 54, 69–74, 76, 79–85, 102, 108, 114, 119, 122, 123

g The function that maps each point of P to its maximum power generation in Microgrid Cable Layout
Used on pages 69, 70, 72–75, 81, 82, 84, 85

$\mathcal{H}$ An expansion
Used on pages 19, 27, 28, 30, 31, 34, 35, 38, 44, 46–49, 51, 53, 55, 62, 63

H The graph of an expansion
Used on pages 19, 46

$H(\cdot)$ The plane graphs representing the lower and upper hull of a diagram A_i
Used on pages 126–134

h The criticality parameter, which we set to 0.614 [Wit+16]
Used on pages 47–51, 53, 56, 152–154

Intervals(S) The set of intervals with endpoints in S, e.g., Intervals($\mathbb{Q}$) is the set of all real intervals with rational endpoints.
Used on pages 10, 11, 19, 102

$\ell_{\max}$ The maximum distance of two poles in a microgrid cable layout
Used on pages 72–74

ℓ The function mapping an edge to its length
Used on pages 69, 71, 72, 74–76, 82–85

$\mathcal{N}$ A flow network
Used on pages 10–15, 17, 19, 27, 28, 30–37, 44, 50, 51, 53, 54, 105–111, 114, 119, 121, 122

$\mathcal{N}(\alpha)$ The flow network corresponding to a node α of an sp-tree
Used on pages 120, 121

$\mathcal{N}^{\text{res}}$ The residual flow network
Used on pages 46–52

nnz The number of non-zero entries of a matrix
Used on page 12

P The set of points given as input to Microgrid Cable Layout
Used on pages 69, 70, 72–74, 79, 81, 82, 85

$P(A_i)$ The projection of the diagram A_i to the xy-plane defined by $P(A_i) = \{(x, y) \in \mathbb{R}^2 \mid \exists \Delta \in \mathbb{R} \colon (x, y, \Delta) \in A_i\}$
Used on pages 124–127, 129, 130

$P_+(A_i)$ The projection of $A_i \cap \left(\mathbb{R}^2 \times [0, \infty)\right)$ to the xy-plane, which is equal to $\{(x, y) \mid \Delta_i^{\max}(x, y) \geq 0\}$
Used on page 133

$P_-(A_i)$ The projection of $A_i \cap \left(\mathbb{R}^2 \times (-\infty, 0]\right)$ to the xy-plane, which is equal to $\{(x, y) \mid \Delta_i^{\min}(x, y) \leq 0\}$
Used on page 133

$P_0(A_i)$ The projection of $A_i \cap \left(\mathbb{R}^2 \times \{0\}\right)$ to the xy-plane, which is equal to $\{(x, y) \mid \Delta_i^{\min}(x, y) \leq 0 \leq \Delta_i^{\max}(x, y)\}$
Used on pages 132–134

r The function that maps the edges to their resistances
Used on pages 16, 71, 82, 83

ρ The function that maps the cable types to their resistivities
Used on pages 69, 71, 73–76, 82, 83

rank The rank of a matrix
Used on page 12

$\mathrm{rev}(e)$	The reverse of the edge e Used on pages 9, 13, 14
S	The set of Steiner points in a microgrid layout Used on page 79
$\mathcal{T}$	An sp-tree Used on pages 21, 35, 119–123
T	The set of timestamps Used on pages 44–48, 51, 53–55, 151–154
τ	A timestamp Used on pages 44–48, 51, 53–55, 151–154
θ	The function that maps vertices to their corresponding angles Used on pages 16–19, 31, 45, 102, 106, 109–111, 114, 124–126, 151–154
$\mathrm{total}(f)$	The total flow value of a flow f, which is defined as $\sum_{v \in V} \max\{\sum_{u:uv \in E} f(uv) - \sum_{w:vw \in E} f(vw), 0\}$. Used on pages 10, 111
U	The voltage of the grid given as input to Microgrid Cable Layout Used on pages 71–73, 75–77, 85
V	The set of vertices of a graph Used on pages 9–13, 16–19, 28, 31, 44–46, 52, 54, 55, 69–75, 79–82, 84, 85, 102, 108–110, 114, 151–154
V_{I}	The set of intermediate vertices of a flow network Used on page 12
V^{res}	The vertices of a residual network Used on page 46
V_{S}	The set of sources of a flow network Used on pages 12–15, 106–108, 110
V_{T}	The set of sinks of a flow network Used on pages 12–15, 106, 108, 110
$\mathrm{viol}(E)$	The total capacity violation of the expansion with edge set E Used on page 55
$y_A^{\max}$	A boundary function of a reduced diagram A defined by $y_A^{\max}(\Delta) = \max\{y \mid (y, \Delta) \in A\}$ Used on pages 115–118
$y_A^{\min}$	A boundary function of a reduced diagram A defined by $y_A^{\min}(\Delta) = \min\{y \mid (y, \Delta) \in A\}$ Used on pages 115–118

A Appendix: Expanding Electrical Networks to Prevent Critical Edges

A.1 The MILP for CC-TNEP

Below we present the MILP for CC-TNEP from Sections 4.3.3 and 4.3.4 in its entirety. The presentation of the constraints has been slightly modified. In particular, we explicitly include the timestamps as subscripts in variables that are time-dependent, and Equation (4.6) is split into two.

$$\text{Minimize} \quad \sum_{e \in E_{\text{cand}}} z(e) \cdot c(e) \tag{4.7}$$

such that

$$d_\tau(v) = \sum_{uv \in E} f_\tau(uv) - \sum_{vw \in E} f_\tau(vw) \qquad \forall v \in V, \tau \in T \tag{4.1}$$

$$|f_\tau(e)| \leq \text{cap}(e) \qquad \forall e \in E_{\text{ex}}, \tau \in T \tag{4.2}$$

$$f_\tau(vw) = b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)) \qquad \forall vw \in E_{\text{ex}}, \tau \in T \tag{4.3}$$

$$|f_\tau(e)| \leq \text{cap}(e) \cdot z(e) \qquad \forall e \in E_{\text{cand}}, \tau \in T \tag{4.4}$$

$$f'_\tau(vw) = b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)) \qquad \forall vw \in E_{\text{cand}}, \tau \in T \tag{4.6a}$$

$$|f_\tau(e) - f'_\tau(e)| \leq M_e \cdot (1 - z(e)) \qquad \forall e \in E_{\text{cand}}, \tau \in T \tag{4.6b}$$

$$f^{\text{res}}_{e,\tau}(e) = 0, \qquad \forall e \in E, \tau \in T \tag{4.8}$$

$$f^{\text{res}}_{e,\tau}(e') \leq \text{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\text{ex}}, \tau \in T \tag{4.9}$$

$$f^{\text{res}}_{e,\tau}(e') \geq -\text{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\text{ex}}, \tau \in T \tag{4.10}$$

$$f^{\text{res}}_{e,\tau}(e') \leq 2\text{cap}(e') \cdot z(e') \qquad \forall e \in E, e' \in E_{\text{cand}}, \tau \in T \tag{4.13}$$

$$f^{\text{res}}_{e,\tau}(e') \geq -2\text{cap}(e') \cdot z(e') \qquad \forall e \in E, e' \in E_{\text{cand}}, \tau \in T \tag{4.14}$$

$$f^{\text{res}}_{e,\tau}(e') \leq \text{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\text{cand}}, \tau \in T \tag{4.15}$$

$$f^{\text{res}}_{e,\tau}(e') \geq -\text{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\text{cand}}, \tau \in T \tag{4.16}$$

$$0 = \sum_{xu \in E} f^{\text{res}}_{vw,\tau}(xu) - \sum_{uy \in E} f^{\text{res}}_{vw,\tau}(uy) \qquad \forall vw \in E, u \in V \setminus \{v, w\}, \tau \in T \tag{4.17}$$

$$c_\tau(e) \geq 0 \qquad \forall e \in E, \tau \in T \tag{4.18}$$

$$c_\tau(vw) \geq f_\tau(vw) - h \cdot \left(\sum_{xw \in E} f^{\text{res}}_{vw,\tau}(xw) - \sum_{wy \in E} f^{\text{res}}_{vw,\tau}(wy) \right) \qquad \forall vw \in E, \tau \in T \tag{4.19}$$

$$c_\tau(vw) \geq -f_\tau(vw) - h \cdot \left(\sum_{xv \in E} f^{\text{res}}_{vw,\tau}(xv) - \sum_{vy \in E} f^{\text{res}}_{vw,\tau}(vy) \right) \qquad \forall vw \in E, \tau \in T \tag{4.20}$$

$$\sum_{\tau \in T} \sum_{e \in E} c_\tau(e) \leq \text{Crit}_{\max}, \tag{4.21}$$

$$f_\tau(e) \in \mathbb{R} \qquad \forall e \in E, \tau \in T$$

$$f'_\tau(e) \in \mathbb{R} \qquad \forall e \in E_{\text{cand}}, \tau \in T$$

$$\theta_\tau(v) \in \mathbb{R} \qquad \forall v \in V, \tau \in T$$

$$c_\tau(e) \in [0, \infty) \qquad \forall e \in E, \tau \in T$$

$$f^{\text{res}}_{e,\tau}(e') \in \mathbb{R} \qquad \forall e \in E, e' \in E, \tau \in T$$

$$z(e) \in \{0, 1\} \qquad \forall e \in E_{\text{cand}}$$

A.2 The MILP for CME

Below we present the MILP for CME from Sections 4.3.3 and 4.3.4 in its entirety. The presentation of the constraints has been slightly modified. In particular, we explicitly include the timestamps as subscripts in variables that are time-dependent, and Equation (4.6) is split into two.

$$\text{Minimize} \quad \sum_{\tau \in T} \sum_{e \in E} c_\tau(e) \tag{4.23}$$

such that

$$d_\tau(v) = \sum_{uv \in E} f_\tau(uv) - \sum_{vw \in E} f_\tau(vw) \qquad \forall v \in V, \tau \in T \tag{4.1}$$

$$|f_\tau(e)| \leq \mathrm{cap}(e) \qquad \forall e \in E_{\mathrm{ex}}, \tau \in T \tag{4.2}$$

$$f_\tau(vw) = b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)) \qquad \forall vw \in E_{\mathrm{ex}}, \tau \in T \tag{4.3}$$

$$|f_\tau(e)| \leq \mathrm{cap}(e) \cdot z(e) \qquad \forall e \in E_{\mathrm{cand}}, \tau \in T \tag{4.4}$$

$$f'_\tau(vw) = b(vw) \cdot (\theta_\tau(v) - \theta_\tau(w)) \qquad \forall vw \in E_{\mathrm{cand}}, \tau \in T \tag{4.6a}$$

$$|f_\tau(e) - f'_\tau(e)| \leq M_e \cdot (1 - z(e)) \qquad \forall e \in E_{\mathrm{cand}}, \tau \in T \tag{4.6b}$$

$$f^{\mathrm{res}}_{e,\tau}(e) = 0, \qquad \forall e \in E, \tau \in T \tag{4.8}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \leq \mathrm{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\mathrm{ex}}, \tau \in T \tag{4.9}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \geq -\mathrm{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\mathrm{ex}}, \tau \in T \tag{4.10}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \leq 2\mathrm{cap}(e') \cdot z(e') \qquad \forall e \in E, e' \in E_{\mathrm{cand}}, \tau \in T \tag{4.13}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \geq -2\mathrm{cap}(e') \cdot z(e') \qquad \forall e \in E, e' \in E_{\mathrm{cand}}, \tau \in T \tag{4.14}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \leq \mathrm{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\mathrm{cand}}, \tau \in T \tag{4.15}$$

$$f^{\mathrm{res}}_{e,\tau}(e') \geq -\mathrm{cap}(e') - f_\tau(e') \qquad \forall e \in E, e' \in E_{\mathrm{cand}}, \tau \in T \tag{4.16}$$

$$0 = \sum_{xu \in E} f^{\mathrm{res}}_{vw,\tau}(xu) - \sum_{uy \in E} f^{\mathrm{res}}_{vw,\tau}(uy) \qquad \forall vw \in E, u \in V \setminus \{v, w\}, \tau \in T \tag{4.17}$$

$$c_\tau(e) \geq 0 \qquad \forall e \in E, \tau \in T \tag{4.18}$$

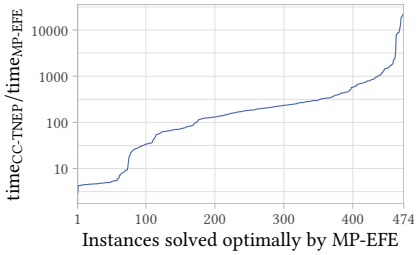
$$c_\tau(vw) \geq f_\tau(vw) - h \cdot \left(\sum_{xw \in E} f^{\mathrm{res}}_{vw,\tau}(xw) - \sum_{wy \in E} f^{\mathrm{res}}_{vw,\tau}(wy) \right) \qquad \forall vw \in E, \tau \in T \tag{4.19}$$

$$c_\tau(vw) \geq -f_\tau(vw) - h \cdot \left(\sum_{xv \in E} f^{\text{res}}_{vw,\tau}(xv) - \sum_{vy \in E} f^{\text{res}}_{vw,\tau}(vy) \right) \qquad \forall vw \in E, \tau \in T \tag{4.20}$$

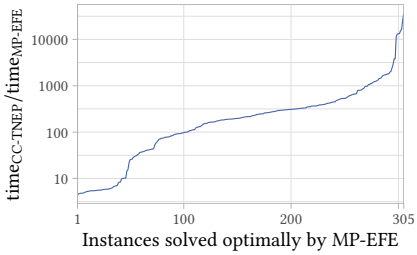
$$\sum_{e \in E_{\text{cand}}} z(e) \cdot c(e) \leq \text{Cost}_{\max} \tag{4.22}$$

$$\begin{aligned}
f_\tau(e) &\in \mathbb{R} && \forall e \in E, \tau \in T \\
f'_\tau(e) &\in \mathbb{R} && \forall e \in E_{\text{cand}}, \tau \in T \\
\theta_\tau(v) &\in \mathbb{R} && \forall v \in V, \tau \in T \\
c_\tau(e) &\in [0, \infty) && \forall e \in E, \tau \in T \\
f^{\text{res}}_{e,\tau}(e') &\in \mathbb{R} && \forall e \in E, e' \in E, \tau \in T \\
z(e) &\in \{0, 1\} && \forall e \in E_{\text{cand}}
\end{aligned}$$

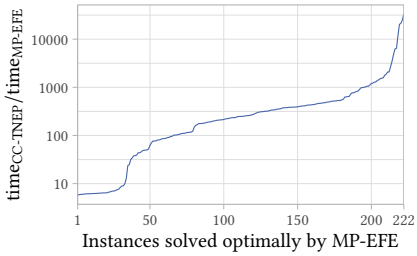
A.3 Plots for CC-TNEP vs. MP-EFE



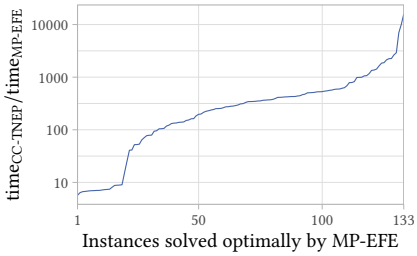
(a) Two timestamps.



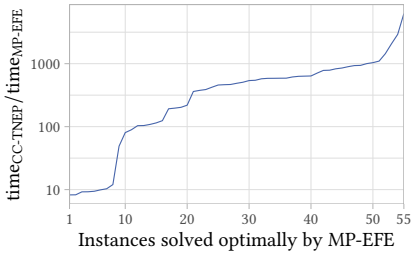
(b) Three timestamps.



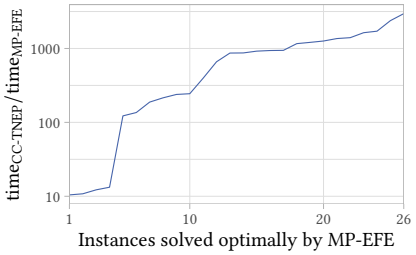
(c) Four timestamps.



(d) Six timestamps.



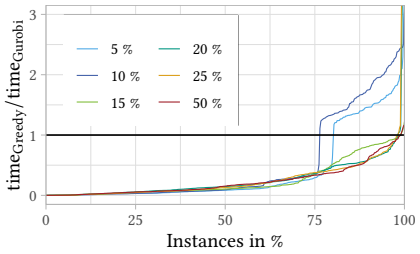
(e) Twelve timestamps.



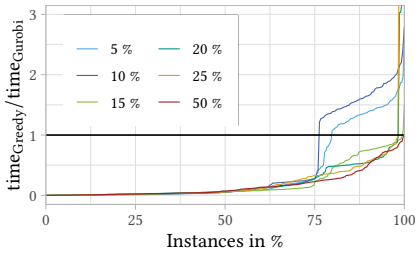
(f) 24 timestamps.

Figure A.1: The ratio of the solution times for CC-TNEP to MP-EFE on instances where both MP-EFE and CC-TNEP found solutions. Each plot shows the results for instances with a fixed number of timestamps. The instances are sorted by increasing ratio.

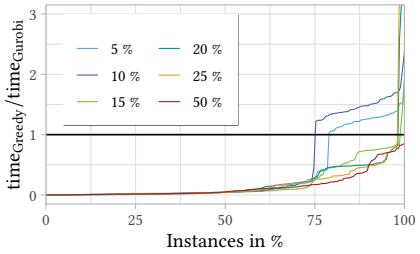
A.4 Plots for the Evaluation of the Greedy Heuristic



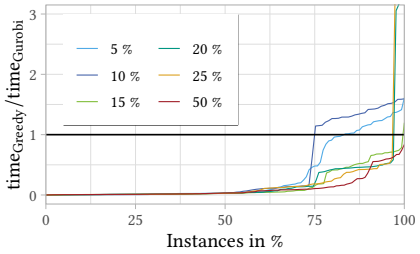
(a) Two timestamps.



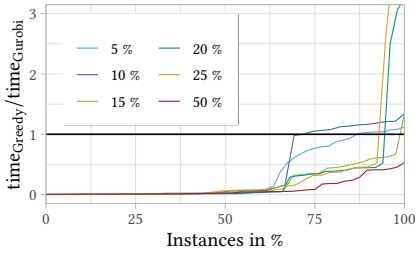
(b) Three timestamps.



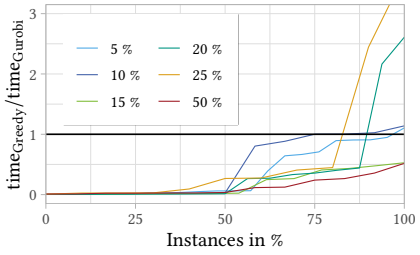
(c) Four timestamps.



(d) Six timestamps.



(e) Twelve timestamps.



(f) 24 timestamps.

Figure A.2: The ratio of solution times for the greedy algorithm and Gurobi solving CME. Only the instances where both solution methods found the same result are plotted. Each plot shows instances with the same number of timestamps. For each budget, the instances are sorted by increasing ratio.