Search Strategies for Topological Network Optimization

Michael D. Moffitt

Google
moffitt@google.com

Abstract

We consider an application of combinatorial search to the optimization of topologies in *series-parallel networks*. We propose a recursive search over the space of *decomposition trees*, in which partial solutions are obtained by exploring k-way partitionings of expandable nodes. We present two complementary pruning techniques that bound the value of intermediate solutions from above and below, applying monotonic operations to the contents of unresolved leaves. We also develop a means to exploit the *convexity* of our objective function, so as to prevent the redundant recomputation of subcircuit configurations. Finally, we evaluate our approach on a parameterized benchmark suite of electrical circuits, demonstrating over an *order of magnitude* improvement in performance as compared to a baseline implementation.

Introduction

There exists a deep and widely-understood relationship between the subjects of *combinatorial optimization* and *network theory*. Their affinity is perhaps best exemplified by *constraint networks* (Dechter 2003), in which the values of a finite set of variables (the *nodes*) interact by way of connections (the *edges*) that restrict their mutual assignments. Similar structures are employed in the neighboring topics of *boolean satisfiability* (Zhang and Malik 2002) and *mixed integer-linear programming* (Achterberg 2009). Despite subtle differences in how constraints and domains are modeled, all of these formulations share a common trait: the network serves as a *static specification*, its contents used by search as a roadmap when exploring the space of solutions.

An entirely different perspective is taken in the lesser-known subject of *topological network design* (Abd-El-Barr 2009; Gupta and Könemann 2011), where the objective is to *construct* – rather than *consume* – a network. Unlike traditional graph problems that can be easily solved in polynomial time (e.g., shortest paths, spanning trees, etc.), the majority of network design problems are NP-hard. This has led previous authors to adopt one of three methodologies: *exhaustive enumeration* when optimally solving very small problems, *iterative refinement* when (suboptimally) solving problems of moderate size, and *approximation algorithms* for some special cases (e.g., trees and acyclic graphs).

In this paper, we consider a radically different algorithmic foundation for the topological optimization of networks. Our approach embraces the principles of advanced combinatorial search, where optimal solutions can be obtained *without* resorting to brute-force enumeration. We argue that the optimization of networks in particular is a formidable problem to tackle, due in part to how solutions are typically evaluated, and also due to the exponential explosion of candidate topologies. Our work focuses on a specific family of *series-parallel* networks that commonly arise in electrical and telecommunications systems. We propose a recursive search over the space of *decomposition trees*, in which partial solutions are obtained by exploring k-way partitionings of expandable nodes. We present two complementary pruning techniques that bound the value of intermediate solutions from above and below, applying monotonic operations to the contents of unresolved leaves. We also develop a means to exploit the *convexity* of our objective function, so as to prevent the redundant recomputation of subcircuit configurations. Finally, we evaluate our approach on a new parameterized benchmark suite, demonstrating over an *order of magnitude* improvement in performance as compared to a baseline implementation.

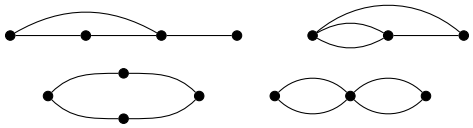
Throughout this manuscript, key concepts are illustrated using simple examples taken from the domain of *electrical circuits*. Our rationale for this is threefold; first, it is one of the earliest applications of network theory, with relevant literature dating back more than a hundred years. Second, the subject matter tends to be more readily accessible to non-specialists, avoiding the kinds of in-depth expertise that complex industrial applications (e.g., supply chain networks, hydrogen transmission pipelines, etc.) might otherwise require. Third, we find that it encapsulates the essential characteristics that make the topic of topological network design a compelling case-study for practitioners in discrete optimization.

Background

The focus of this work lies at the intersection of multiple specialties, including graph theory, electronic design automation, network optimization, and combinatorial search. Indeed, our approach builds upon a wealth of previous contributions to these areas, which we briefly cover in the following sections.

Graph Theory

A network is said to be *series-parallel* if it can be reduced to a single edge through repeated contraction, whereby any degree-2 node is replaced by an edge, and any parallel edges (sharing common endpoints) are collapsed into one. To the best of our knowledge, the study of series-parallel networks as a serious topic for discrete mathematics was first proposed in the late 19th century, motivated by the immense variety of configurations observed in electrical circuits (MacMahon 1892). The sequence of cardinalities in that work considered graphs with *unlabeled* edges only, such as the handful of configurations over four resistors depicted below:



In the decades that followed, the subject of series-parallel networks continued to intrigue graph theorists. The classification of circuit structures (broken down by nullity and rank) prompted one such investigation (Foster 1932). A subsequent study developed rigorous recurrence relations that proved the validity of MacMahon's original sequences (Riordan and Shannon 1942). The case of graphs containing *labeled* edges was eventually addressed (Carlitz and Riordan 1956), resulting in a sequence of much larger values. The relationships between series-parallel topologies and *confluent networks* & *combinatorial pregeometries* were explored in two other seminal works (Duffin 1965; Brylawski 1971). Finally, the presence of *cut-nodes* was used to characterize series-parallel graphs into α- and π-networks (Moon 1987).

Notably, all of the works above focus exclusively on the enumeration or categorization of topologies. Several studies have instead considered *optimization* within a given series-parallel topology (Takamizawa, Nishizeki, and Saito 1982; Bein, Brucker, and Tamir 1985; Hassin and Tamir 1986; Klinz and Woeginger 2004). However, the (arguably) more challenging problem of optimizing across multiple topologies has so far been avoided by theoretical mathematicians.

Electronic Design Automation

The topic of optimization is front-and-center in modern electronics. While some attention has been given to applications in analog circuitry (Liu et al. 2009; Barros, Guilherme, and Horta 2010), most industrial models are instead designed using digital primitives (Lavagno et al. 2016). The computer-aided design of integrated circuits is traditionally broken down into two categories: the former deals with *logical synthesis* (Hachtel and Somenzi 1996), such as technology mapping and equivalence verification. The latter deals with *physical synthesis*, including the placement of gates (Nam and Cong 2007) and routing of wires (Hu and Sapatnekar 2001).

With modern designs containing billions of transistors, the optimization strategies for these "mega-networks" are limited to the realm of suboptimal search. Stochastic techniques, such as simulated annealing and gradient descent, remain the standard for many industrial place & route tools.

Network Optimization

Outside the context of circuits, network optimization is a widely-celebrated topic that spans many areas. In operations research (Bertsekas 1998), it includes a variety of NP-hard problems such as *traveling salesman*, *vehicle routing*, etc. These formulations tend to concentrate on variable assignments within a *fixed* topology; i.e., the labeling of arcs between nodes in a fully-connected graph, or computing pairwise matches between vertices. Even approaches that do explore the generation of topologies – such as *minimum spanning trees* (Cheriton and Tarjan 1976), *path optimization* (Dionne and Florian 1979), and *multicommodity flows* (Balakrishnan, Li, and Mirchandani 2017) – often owe their success to the specific subclasses of networks that they consider.

Real-world applications of network design are numerous, extending to the optimization of transportation costs (Boyce and Soberanes 1979), communication networks (Yang and Ephremides 1997; Cheng 1998; Neely 2010), reliable systems (Jan, Hwang, and Chen 1993; Levitin et al. 1998; Liu and Iwamura 2000), wireless networks (Miao et al. 2009; Pathak and Dutta 2011), supply chains (Nagurney 2010; Eskandarpour et al. 2015), transmission pipelines (André et al. 2013), water distribution networks (Zheng et al. 2013), conductor connections (Nomura et al. 2019), and photovoltaic systems (Kurmanbay et al. 2020). Several of these employ the same class of series-parallel networks that interest us, yet their methodologies are limited to local search due to computational intractability. A survey of topological network design (Abd-El-Barr 2009) reinforces this view, citing only exhaustive enumeration as a means of producing optimal solutions.

Combinatorial Search

Of all the disciplines related to our work, perhaps the most similar in spirit is the family of search strategies commonly used in planning (Ghallab, Nau, and Traverso 2004), scheduling (Pinedo 2016), and constraint processing (Dechter 2003). A key premise that permeates these fields is the principle that optimal solutions to NP-hard problems can typically be obtained *without* resorting to brute-force enumeration. Powerful inference and pruning techniques – including arc consistency, backjumping, branch-and-bound, etc. – can be combined to make the tractable exploration of solutions possible (Russell and Norvig 2020).

In many cases, a network will serve as the *input* to a solver (e.g., a constraint network). These networks can even represent digital circuits (Fattah and Dechter 1995). However, rarely will a network be the *output*, with the possible exception of planning in *hierarchical task networks* (Erol, Hendler, and Nau 1994; Nau et al. 2003). Here, partial orderings over primitive tasks are produced that honor conditional dependencies among actions.

Progress in combinatorial search is routinely measured on standardized benchmarks. The evaluation of our work relies upon a *parameterized* suite, inspired by previous studies into the packing of squares 1×1 to $n\times n$ (Korf, Moffitt, and Pollack 2010) and the partitioning of numbers between 1 and n (Korf 1998). In fact, our problem is closely related to this latter formulation, as discussed in the following section.

Problem Definition

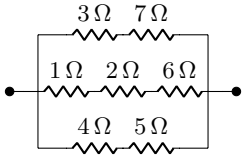
We define our network optimization problem as a tuple $\langle \mathcal{S}, f_+, f_\parallel, f_T \rangle$, where $\mathcal{S}$ is a multiset of numeric labels, f_+ and $f_\parallel$ are monotone operators for series and parallel connections (respectively), and f_T is a target network value. The objective is to construct a series-parallel network $\mathcal{N}$ from all the elements in $\mathcal{S}$ that minimizes the cost $|f(\mathcal{N}) - f_T|$, with $f(\mathcal{N})$ being the value of network $\mathcal{N}$.

Application to Electrical Circuits

To illustrate our formulation, we turn to a straightforward application in the context of electrical circuits (Alexander and Sadiku 2016), where resistors in series combine additively, and resistors in parallel combine reciprocally:

$$f_+(\mathcal{S}) = \sum_{s_i \in \mathcal{S}} s_i \qquad f_\parallel(\mathcal{S}) = \left(\sum_{s_i \in \mathcal{S}} {}^1\!/_{s_i} \right)^{-1}$$

In order to achieve a desired total resistance, raw components – whose allowable values are invariably predetermined by industry standards (for inventory simplification) – must be creatively combined. For example, consider a simple 7-resistor example where $\mathcal{S} = \{1, 2, ..., 7\}$ and $f_T = \sqrt{7}$. One solution (albeit suboptimal) is the following circuit:



In this topology, we observe exactly three series connections, whose local resistance values (in ohms) are $\langle 10, 9, 9 \rangle$ from top to bottom. When these are combined in parallel, the total resistance can be found to equal $({}^1\!/_{10} + {}^1\!/_9 + {}^1\!/_9)^{-1}$ ohms, or equivalently, ${}^{90}\!/_{29}\ \Omega$. Given a target resistance of $\sqrt{7}$, the absolute difference is approximately 0.4577.

Similarities to Number Partitioning

The circuit above can be viewed as a partitioning over the elements in $\mathcal{S}$:

$$\mathcal{P} = \{\{3, 7\}, \{1, 2, 6\}, \{4, 5\}\}$$

Since any partitioning of resistors can be translated into a valid series-parallel circuit, our formulation bears some resemblance to the *number partitioning* problem (Korf 2009, 2011), which seeks to divide a set of n elements across k partitions $\{S_1, S_2, ..., S_k\}$ so as to minimize the maximum subset sum:

$$\max_i \left(\sum_{s_{ij} \in S_i} s_{ij} \right)$$

Our partitioning $\mathcal{P}$ turns out to be an *optimal* solution for the case where $k = 3$, since the maximum subset sum of 10 cannot be reduced.

Despite the superficial similarity between these problems, we contend that the task of network optimization presents at least two distinct challenges.

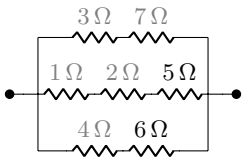
Sensitivity to Solution Structure

The first challenge relates to the sensitivity of the objective function to small changes in solution structure. Number partitioning has been shown to exhibit *weakest-link optimality* (Moffitt 2013), a property that allows globally optimal complete solutions to be constructed from locally *suboptimal* partial solutions. Modern implementations exploit this principle by relaxing upper and lower bounds on solution quality when performing recursive decomposition (Schreiber, Korf, and Moffitt 2018).

When applied to our running example, we observe that the partitioning $\mathcal{P}$ contains two subsets whose sums are each 9, and thus do not influence the total solution cost of 10. As a result, one can easily shuffle subset contents without affecting solution quality, such as in the following alternate (and also optimal) partitioning of the elements in $\mathcal{S}$:

$$\mathcal{P}' = \{\{3, 7\}, \{1, 2, 5\}, \{4, 6\}\}$$

Unfortunately, the same cannot be said for its series-parallel circuit equivalent, illustrated below:

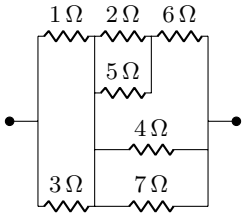


The total resistance of this new solution can be found to equal $({}^1\!/_{10} + {}^1\!/_8 + {}^1\!/_{10})^{-1}$ ohms, or equivalently, ${}^{40}\!/_{13}\ \Omega$. Again, assuming a target resistance of $\sqrt{7}$, the absolute difference is approximately 0.4312, demonstrating a small improvement in cost over our initial circuit.

This higher sensitivity has the potential to require more computational effort (as compared to number partitioning) when exploring the space of possible solutions.

Size of Solution Space

The difficulty described in the previous section is compounded by a second major challenge: the solution space of viable network topologies is overwhelmingly large. For example, consider the following series-parallel circuit:



The recursively nested structures in this solution have no analogous counterpart in the (comparatively flatter) number partitioning formulation. Furthermore, because the presence of branching subcircuits provides an increase in topological flexibility, high-quality solutions tend to make liberal use of them. Indeed, the above circuit is optimal; we leave it as an exercise to the reader to verify that its total resistance is ${}^{127}\!/_{48}\ \Omega$, a value that lies remarkably close to our target:

$$\sqrt{7} = \mathbf{2.645}751...$$
$${}^{127}\!/_{48} = \mathbf{2.645}833...$$

Exactly *how* much larger is the solution space? For each value of n in $\{10, 20, 30\}$, we compare the number of distinct partitionings over n labels to the number of series-parallel networks containing n elements:[1]

n	# of partitionings	# of networks
10	1.159×10^5	5.642×10^8
20	5.172×10^{13}	1.774×10^{24}
30	8.467×10^{23}	1.414×10^{42}

For the largest of these values, the number of viable networks is nearly *twenty orders of magnitude greater* than the number of partitionings.

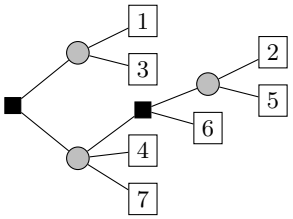
Our Approach

In this section, we outline our basic approach to the topological optimization of series-parallel networks. It can be characterized as *exact* (i.e., producing optimal solutions), *anytime* (i.e., providing intermediate results), and *fast* (i.e., avoiding exhaustive enumeration).

Solution Representation

To encode solutions, we make use of *decomposition trees* (Valdes, Tarjan, and Lawler 1982; Lengauer 1990), a recursive structure that represents series and parallel connections as alternating layers of intermediate nodes (with terminating leaf nodes signifying individual elements). This mirrors an approach taken in *factored planning* (Kelareva et al. 2007), where a similar container specifies partitions of domains into subdomains. They are also cosmetically comparable to the AND/OR trees prevalent in optimization for graphical models (Dechter and Mateescu 2007; Marinescu and Dechter 2009), with the obvious exception that we use them to capture network topologies rather than search spaces.

The decomposition tree for the optimal solution presented earlier is shown below, using squares and circles to indicate series and parallel connections, respectively:



For partial solutions, we permit leaves to contain element subsets; we call these *expandable nodes*. Each subset represents a subcircuit whose structure has yet to be resolved.

Recursive Decomposition

Given a decomposition tree $\mathcal{T}$, we select some expandable node $\mathcal{S}_0 \subseteq \mathcal{S}$ and for every $k > 1$ explore all possible k-way partitionings $\mathcal{P}(\mathcal{S}_0)$ over its elements. The resulting decomposition tree $\mathcal{T}'$ for each partitioning is then expanded (depth-first) until no expandable nodes remain; such leaf nodes represent complete solutions. Hence, our approach performs a recursive search over nodes that *themselves* represent recursive decompositions of the network elements.

[1]These correspond to sequences A000110 and A006351, respectively, in the OEIS database (Sloane 1991).

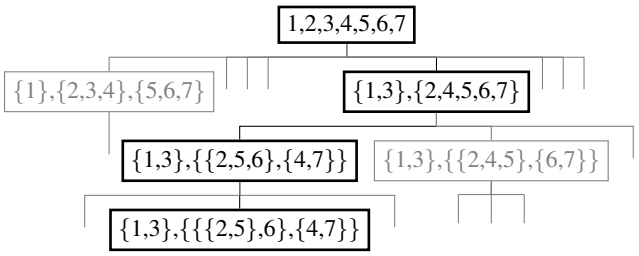
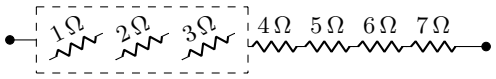


Figure 1: Exploring the space of decomposition trees, with the path to an optimal solution highlighted by nodes in dark.

Bounding Conditions

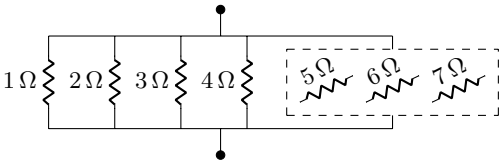
While an exhaustive search through the space of decomposition trees (illustrated in Figure 1) produces optimal solutions, it is nevertheless impractical; as discussed in previous sections, the number of such trees is prohibitively large.

Fortunately, partial solutions may be pruned in two ways. For any monotone operators $f_\parallel$ and f_+ such that $f_\parallel(\mathcal{S}) \leq f_+(\mathcal{S})\ \forall \mathcal{S}$, we can first obtain a *lower bound* $f_L(\mathcal{N})$ on the value of any partial solution $\mathcal{N}$ by applying $f_\parallel$ to the elements of each expandable node.[2] If the value $f_L(\mathcal{N})$ is sufficiently large – i.e., larger than $f_T + c$ (where c is the cost of the best known solution) – the partial solution may be abandoned.



Example: Consider the partial solution above, in which resistors $4\ \Omega$ through $7\ \Omega$ have been placed in series. Even if the remaining elements $1\ \Omega$ through $3\ \Omega$ were all to be placed in parallel, the resulting total resistance $f_L(\mathcal{N})$ would still be $^{248}/_{11}\ \Omega$. Hence, if a solution of cost c is known such that $c \leq {}^{248}/_{11} - \sqrt{7}$, this partial solution may be pruned. □

Similarly, we can also obtain an *upper bound* $f_U(\mathcal{N})$ by instead applying f_+ to the elements of each expandable node. One may safely backtrack from this solution if $f_U(\mathcal{N})$ is smaller than $f_T - c$.



Example: Consider the partial solution above, in which resistors $1\ \Omega$ through $4\ \Omega$ have been placed in parallel. Even if the remaining elements $5\ \Omega$ through $7\ \Omega$ were all to be placed in series, the resulting total resistance $f_U(\mathcal{N})$ could not exceed $^{36}/_{77}\ \Omega$. Hence, if a solution of cost c is known such that $c \leq \sqrt{7} - {}^{36}/_{77}$, this partial solution may be pruned. □

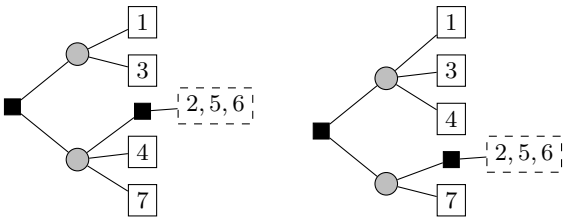
[2]This calculation involves a single traversal through the decomposition tree, and thus exhibits a time complexity of $O(n)$.

Exploiting Convexity with Tabulation

During the exploration of decomposition trees, it is inevitable that many subsets of network elements will be encountered at multiple expandable nodes. In this section, we reveal two techniques that can significantly help to avoid redundant recomputation of such subproblems.

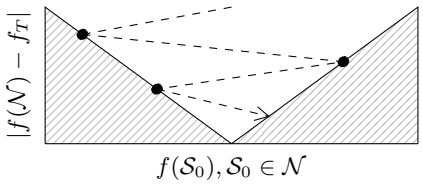
Case #1: Singleton Expandables

Consider the two partial solutions below, both of which contain a solitary expandable node:



In each tree, the subset $\mathcal{S}_0 = \{2, 5, 6\}$ must be recursively expanded so as to find the best possible configuration. On the one hand, an optimal subconfiguration for one tree is not guaranteed to be optimal for the other; depending on the topology of the surrounding network, the local value of $f(\mathcal{S}_0)$ may need to be higher or lower. Yet, many of the same solutions will be repeatedly recomputed from scratch whenever subproblems are of sufficient similarity.

Fortunately, our objective function $|f(\mathcal{N}) - f_T|$ happens to be *convex* with respect to the value $f(\mathcal{S}_0)$ of any individual subcircuit $\mathcal{S}_0 \in \mathcal{N}$. In other words, the global cost of the network is guaranteed to increase or decrease monotonically as a function of $f(\mathcal{S}_0)$ when $\mathcal{S}_0$ is considered in isolation:



While convexity can be exploited in a variety of ways (Bertsekas 2015), the approach we take is tailored to our decomposition tree. Specifically, when only a *single* expandable node $\mathcal{S}_0$ remains in $\mathcal{T}$, we perform a simple binary search over its possible configurations – denoted $\mathcal{C}(\mathcal{S}_0)$ – sorted by their respective network values in a dedicated lookup table.[3]

Example: Consider the leftmost partial solution above. For node $\mathcal{S}_0 = \{2, 5, 6\}$, its various configurations $\mathcal{C}(\mathcal{S}_0)$ are given in Table 1(a).[4] The entire top half of these subcircuits can be rejected as candidates immediately, since even the largest of these ($^{42}/_{13}\ \Omega$) would lead to a global resistance value far less than the target. By continuing the binary search, we ultimately select the subcircuit with resistance $^{52}/_{7}\ \Omega$. □

[3]This requires some additional precomputation, and because the number of configurations grows exponentially, the approach is restricted to subset cardinalities below a certain maximum size $\mathcal{M}$.

[4]Here and in future sections, we express decomposition trees using infix notation – e.g., (1|3)+(((2|5)+6)|4|7) – to save space. The symbols '+' and '|' refer to f_+ and $f_\parallel$, respectively.

Subcircuit	Resistance
2|5|6	$^{15}/_{13}\ \Omega$
2|(5+6)	$^{22}/_{13}\ \Omega$
(2+6)|5	$^{40}/_{13}\ \Omega$
(2+5)|6	$^{42}/_{13}\ \Omega$
2+(5|6)	$^{52}/_{11}\ \Omega$
(2|6)+5	$^{13}/_{2}\ \Omega$
(2|5)+6	$^{52}/_{7}\ \Omega$
2+5+6	$13\ \Omega$

(a) $\mathcal{C}(\{2, 5, 6\})$

Subcircuit	Resistance
1|3|4	$^{12}/_{19}\ \Omega$
1|(3+4)	$^{7}/_{8}\ \Omega$
(1+4)|3	$^{15}/_{8}\ \Omega$
(1+3)|4	$2\ \Omega$
1+(3|4)	$^{19}/_{7}\ \Omega$
(1|4)+3	$^{19}/_{5}\ \Omega$
(1|3)+4	$^{19}/_{4}\ \Omega$
1+3+4	$8\ \Omega$

(b) $\mathcal{C}(\{1, 3, 4\})$

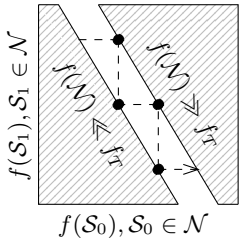
Table 1: Subcircuit configurations sorted by resistance.

Case #2: Expandable Pairs

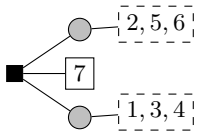
When exactly two expandable nodes $\mathcal{S}_0$ and $\mathcal{S}_1$ remain, our task is more difficult: while $|f(\mathcal{N}) - f_T|$ may be convex with respect to $f(\mathcal{S}_0)$ or $f(\mathcal{S}_1)$ independently, the joint solution space of both subproblems must be considered in tandem.

Even for this case, we can still leverage convexity in the following way: beginning with $f_L(\mathcal{S}_0)$ and $f_U(\mathcal{S}_1)$, we perform a dual linear sweep over the space of pairwise configurations:

- If the value of resulting network $f(\mathcal{N})$ happens to be smaller than our target f_T, we consider a subcircuit for $\mathcal{S}_0$ with the next *higher* value in our lookup table.
- Otherwise, we consider a new subcircuit for $\mathcal{S}_1$ instead, taking the entry with the next *lower* value.



By continuing the sweep, our approach reduces the number of candidate solutions from $|\mathcal{C}(\mathcal{S}_0)| \times |\mathcal{C}(\mathcal{S}_1)|$ (in the worst case) to $|\mathcal{C}(\mathcal{S}_0)| + |\mathcal{C}(\mathcal{S}_1)|$, essentially navigating a narrow corridor of potentially cost-minimizing networks. This technique (whose proof of optimality is given in our Appendix) mimics a landmark improvement to algorithms for the knapsack problem (Horowitz and Sahni 1974), where lookup tables are instead used to retrieve partial subset sums.



Example: Consider the partial solution above. For nodes $\mathcal{S}_0 = \{2, 5, 6\}$ and $\mathcal{S}_1 = \{1, 3, 4\}$, their various configurations $\mathcal{C}(\mathcal{S}_0)$ and $\mathcal{C}(\mathcal{S}_1)$ are given in Tables 1(a) and 1(b), respectively. The smallest resistance for the former is $^{15}/_{13}\ \Omega$, whereas the largest resistance for the latter is $8\ \Omega$. Since a circuit composed of these configurations exceeds the target resistance, we would consider the next *lower* subcircuit for $\mathcal{S}_1$ with resistance $^{19}/_{4}\ \Omega$. □

Algorithm : NETWORK-OPT

global vars $(\mathcal{N}, c) \leftarrow (\varnothing, \infty)$ (the best network and its cost)
Input: A decomposition tree $\mathcal{T}$, initially a singular node ($\mathcal{S}$)
Parameter: $\mathcal{M}$ (maximum subset size for tabulation)
Output: A network $\mathcal{N}$ that minimizes $|f(\mathcal{N}) - f_T|$

1: **if** $f_L(\mathcal{T}) \geq f_T + c$ **or** $f_U(\mathcal{T}) \leq f_T - c$ **then**
2: **return**
3: **end if**
4: $\{\mathcal{S}_0, \mathcal{S}_1, \mathcal{S}_2, ...\} \leftarrow$ EXPANDABLE-NODES($\mathcal{T}$)
5: **if** $\mathcal{S}_0 = \varnothing$ **then**
6: $(\mathcal{N}, c) \leftarrow (\mathcal{T}, |f(\mathcal{T}) - f_T|)$
7: **else if** $\mathcal{S}_1 = \varnothing$ **and** $|\mathcal{S}_0| \leq \mathcal{M}$ **then**
8: $\mathcal{T}' \leftarrow \mathcal{T} \cup$ BINARY-SEARCH$(\mathcal{T}, \mathcal{S}_0) \setminus \mathcal{S}_0$
9: NETWORK-OPT($\mathcal{T}'$)
10: **else if** $\mathcal{S}_2 = \varnothing$ **and** $\max(|\mathcal{S}_0|, |\mathcal{S}_1|) \leq \mathcal{M}$ **then**
11: $\mathcal{T}' \leftarrow \mathcal{T} \cup$ LINEAR-SWEEP$(\mathcal{T}, \mathcal{S}_0, \mathcal{S}_1) \setminus \mathcal{S}_0 \setminus \mathcal{S}_1$
12: NETWORK-OPT($\mathcal{T}'$)
13: **else**
14: **for** $\mathcal{P} \in$ PARTITIONINGS$(\mathcal{S}_0)$ **do**
15: $\mathcal{T}' \leftarrow \mathcal{T} \cup \mathcal{P} \setminus \mathcal{S}_0$
16: NETWORK-OPT($\mathcal{T}'$)
17: **end for**
18: **end if**

The Complete Algorithm

The pseudocode for our topological network optimization algorithm is shown above. The function NETWORK-OPT accepts a decomposition tree $\mathcal{T}$ as input, initialized to a single expandable node containing the set of all network elements $\mathcal{S}$. If either the lower or upper bound on cost establishes this network to be suboptimal, the solution is promptly abandoned (lines 1-3). Otherwise, a $\varnothing$-terminated list of expandable nodes is collected, and one of four cases is considered:

- If no expandable node remains, the solution is complete, and is stored along with its associated cost (lines 4-6). Due to our bounding conditions, this network is guaranteed to improve upon any previously found solutions.
- If exactly one expandable node remains (and is below a given size), we perform a binary search over precomputed subcircuits to retrieve its replacement (lines 7-9).
- If exactly two expandable nodes remain (and are both below a given size), we perform a dual linear sweep over precomputed subcircuits to retrieve their replacements (lines 10-12).
- For any other partial solution, we select a node to expand and iterate over its viable partitionings. We resolve each of these expansions with a recursive call (lines 13-18).

As noted in previous sections, the precomputed subcircuits are obtained from a dedicated lookup table that is generated prior to search; we do not show that code here, but it follows a simple exhaustive enumeration of decomposition trees that closely resembles the approach above.[5] All decomposition trees are constructed incrementally (i.e., a singular structure is maintained across all levels of search) to reduce runtime.

[5]Source code for our implementation (including utilities to reproduce all figures in this paper) is available upon request.

Empirical Results

To evaluate the efficiency of our algorithm, we introduce a benchmark suite containing parameterized electrical circuits of increasing difficulty. For progressively larger values of n, we assign $\mathcal{S}$ and f_T as follows:

$$\mathcal{S} = \{V_1, V_2, ..., V_n\} \qquad f_T = \sqrt{n}$$

where V_i is the i^{th} entry in the E12 series of *preferred values* for electronic components, established seventy years ago as part of an international standard (IEC 60063) that has since been adopted by multiple national committees. The first *decade* of such numbers – designed to be equally spaced along a logarithmic scale – is given below:

$1.0\,\Omega$	$1.2\,\Omega$	$1.5\,\Omega$	$1.8\,\Omega$	$2.2\,\Omega$	$2.7\,\Omega$
$3.3\,\Omega$	$3.9\,\Omega$	$4.7\,\Omega$	$5.6\,\Omega$	$6.8\,\Omega$	$8.2\,\Omega$

Note that for any non-square value of n (where our target f_T is irrational), the objective function $|f(\mathcal{N}) - f_T|$ is guaranteed to admit no "perfect" (i.e., zero-valued) solutions.

In order to demonstrate the relative superiority of our optimal solutions, we have also implemented an entirely separate solver that uses stochastic local search to produce suboptimal networks. This process begins by constructing a topologically randomized network, followed by iterative refinement through the repeated replacement of circuit substructures: pairs of subcircuits of size $\mathcal{M}$ or smaller are selected at random, and locally optimized using our dual linear sweep. Whenever solution quality reaches a plateau, a random restart is employed to reset the global circuit structure. This solver is allowed to run for the same length of time as it takes for our exact solver to complete its search.

The results are shown in Table 2. For each value of n, we provide the costs of both the suboptimal & optimal solutions, along with the optimal circuit itself and the runtime for four different incarnations of our algorithm:[6]

- An exhaustively enumerative baseline (i.e., no pruning).
- A branch-and-bound version that uses $f_L(\mathcal{N})$ and $f_U(\mathcal{N})$ to prune partial solutions.
- A version that exploits convexity with tabulation (we include the time needed to precompute lookup tables).
- Finally, a version that combines all techniques described in this paper.

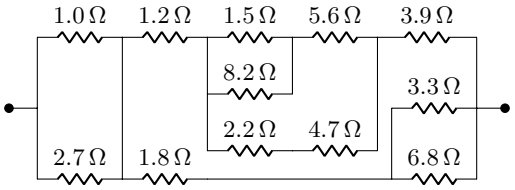
All solvers perform reasonably well for $n = 5$, as it is trivially small. However, the benefits of our techniques become abundantly clear at $n \geq 6$; compared to the baseline, our algorithm is up to $\underline{\mathbf{45}}\times$ faster. Furthermore, it appears that *both* of our key enhancements are required to achieve this level of performance, whereas a slowdown of $3\times$ or more is observed when either technique is individually disabled. As for solution quality, the cost of each optimal network tends to be substantially smaller than that produced by the local search implementation, often by a factor of $4\times$ or more.

[6]All experiments were conducted on a Debian Linux workstation powered by a 2.20GHz Intel® Xeon® CPU and 32gb of RAM. A timeout of one week was imposed on all solvers. For any experiment using tabulation, the hyperparameter $\mathcal{M}$ was set to $\lceil n/2 \rceil$.

n	Cost (Subopt.)	Cost (Opt.)	Optimal Circuit	Runtime			
				Baseline	Bounding	Tabulation	**Combined**
5	3.931×10^{-2}	8.830×10^{-3}	(1|((1.2|1.8) ... ... +2.2))+1.5	0.008 sec	0.007 sec	0.006 sec	**0.005 sec**
6	3.161×10^{-3}	1.005×10^{-3}	((1+1.8)|(2.2 ... ... +2.7))+(1.2|1.5)	0.081 sec	0.038 sec	0.047 sec	**0.024 sec**
7	3.528×10^{-4}	9.549×10^{-5}	((1+2.7)|((1.2|2.2) ... ... +3.3)|1.5)+1.8	1.292 sec	0.468 sec	0.243 sec	**0.140 sec**
8	1.587×10^{-5}	1.758×10^{-6}	(1|2.2)+((1.2+1.5 ... ... +1.8+(2.7|3.9))|3.3)	24.666 sec	7.023 sec	4.468 sec	**1.727 sec**
9	2.661×10^{-6}	0 (*"perfect"*)	(((((1|1.8)+4.7)|3.3) ... ... +1.5+2.7)|1.2|3.9)+2.2	10m	2m	1m	**< 1m**
10	2.343×10^{-8}	5.058×10^{-9}	(((1|(1.5+4.7)|2.7|5.6) ... ... +((1.8|3.3)+2.2))|3.9)+1.2	3hr 48m	47m	20m	**7m**
11	3.101×10^{-9}	2.763×10^{-10}	(((1|2.7|3.3)+((((1.2|4.7|5.6) ... ... +3.9)|6.8)+1.5))|2.2)+1.8	103hr 54m	19hr 10m	6hr 34m	**2hr 16m**
12	3.053×10^{-10}	6.060×10^{-12}	(1|2.7)+((1.2+(((1.5|8.2)+5.6) ... ... |(2.2+4.7))+3.9)|(1.8+(3.3|6.8)))	*TIMEOUT*	*TIMEOUT*	*TIMEOUT*	**64hr 30m**

Table 2: Results of our topological network optimization algorithm over electrical circuits containing the E12 series resistors.

At the time of this writing, the largest problem that we are able to solve optimally is that of $n = 12$. We include a schematic of that solution below:



Notably, the resistance of this circuit rationally approximates the target to within eleven decimal places:

$$\sqrt{12} = \mathbf{3.46410161513}775...$$
$$^{12067164184}/_{3483490245} = \mathbf{3.46410161513}169...$$

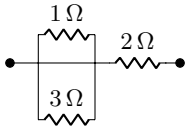
Special Cases

While our problem statement is crafted to be broadly applicable, certain special cases are more likely to be encountered in practice than others. Here, we cover two such variations, and discuss strategies to ensure efficient performance.

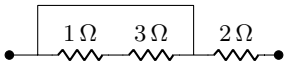
Optional Exclusion of Network Elements

In real-world situations, one reasonable modification is to relax the requirement that all network elements must participate in the solution. For instance, a circuit created from a strict subset of electrical components may produce a better approximation than one which is forced to use all of them.

Perhaps surprisingly, it is possible to repurpose our original formulation to accommodate this task. In order to enable the optional exclusion of resistors, it is sufficient to introduce a single 0 Ω resistor into the multiset $\mathcal{S}$. This essentially serves as a bare wire that allows the solver to "remove" any subset of resistors via *electrical shorting*, whereby $f_{\parallel}(\mathcal{S})$ resolves to 0 for any $\mathcal{S}$ such that $0 \in \mathcal{S}$. As an illustrative example, the contributions of resistors 1 Ω and 3 Ω in the following circuit have both been completely eliminated:



What makes this "0 Ω trick" unusual is that it opens the door to a number of potentially degenerate topologies; e.g., ones where the bare wire is placed at arbitrary depths, or where the relative placement of shorted elements leads to the exploration of electrically redundant circuit structures:



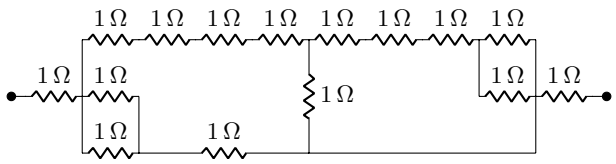
These adverse effects can be mitigated by ensuring that the solver honors two additional constraints:

- Any 0 Ω resistor should be placed at most one level deep.
- Any 0 Ω subcircuit should be at most one element wide (i.e., should contain no nested series connections).

Duplication of Network Elements

Our choice of a multiset (rather than a mere set) for $\mathcal{S}$ is deliberate; given sufficient inventory, there are likely to be many copies of the same network element available for use. Since duplicates are interchangeable, their presence affords additional opportunities for runtime reduction through *symmetry breaking*.

This can be easily achieved by ensuring that the PARTITIONINGS() function avoids repeated elements in the sequence it returns. By incorporating this adaptation, we were able to produce a 15-element circuit comprised entirely of 1 Ω resistors that approximates π to six digits:



$$\pi = \mathbf{3.141592}653...$$
$$^{355}/_{113} = \mathbf{3.141592}920...$$

Future Work

We believe that the study of topological network optimization will continue to be an attractive area for future work. At the very least, we are hopeful to see the set of optimal solutions for our benchmark suite extended to $n = 13$ and beyond.

One possible avenue of research is to broaden the application of tabulation to the case of three or more expandable nodes, although it is not yet clear how this can be done while at the same time taking advantage of convexity. Perhaps a *meta-CSP* – akin to the ones commonly employed in constraint-based temporal reasoning (Tsamardinos and Pollack 2003) – could more efficiently search the space of viable network configurations, especially since series-parallel TCSPs can be solved in polynomial time (Dechter, Meiri, and Pearl 1991).

Second, it may be necessary to abandon the expectation of optimality entirely, and instead pursue algorithms capable of producing high-quality (but suboptimal) solutions quickly. For larger problems containing hundreds of components, the framework of *anytime heuristic search* (Hansen and Zhou 2007) could potentially outperform the iterative refinement techniques previously adopted in network design, assuming that a useful admissible function is available.

Finally, it might be worth investigating search strategies for more exotic circuit structures, e.g. power networks (Boardman and Meckiff 1985; Singh and Sapatnekar 2004) and optical waveguide crossings (Fujimoto et al. 2012). The *reversible circuits* found in quantum computing (Saeedi and Markov 2013; Davis et al. 2020; Itoko et al. 2020) are also small enough that optimal solutions may be within reach.

Conclusion

We have considered the topological optimization of series-parallel networks. Despite the myriad of challenges involved in this task, we have devised a set of search strategies that significantly reduce the computational burden of constructing optimal solutions. Our approach exploits the monotonicity of network operators through the administration of branch-and-bound pruning, as well as the convexity of our objective function through tabulation and accelerated retrieval at the bottom-most levels of search. Together, these techniques are shown to substantially improve the performance of recursive decomposition, as demonstrated across a new suite of parameterized benchmarks. While the efficiency of our algorithm has been illustrated in the context of one specific application – the configuration of electrical circuits – it is also domain agnostic, and can be applied to the design of any network that meets our criteria.

Acknowledgments

The author is especially grateful to Vincent Furnon for providing invaluable feedback on a preliminary version of this work. We also appreciate the commentary provided by the anonymous reviewers, whose thoughtful recommendations were instrumental in improving the quality of this paper.

Appendix: Proof of Optimality

Our dual linear sweep performs a piecewise traversal through a two-dimensional Euclidean subspace. Each coordinate (x, y) in this plane corresponds to some pair $(f(\mathcal{N}_0), f(\mathcal{N}_1))$, where $\mathcal{N}_0 \in \mathcal{C}(\mathcal{S}_0)$ and $\mathcal{N}_1 \in \mathcal{C}(\mathcal{S}_1)$. Rather than explore the entire space – i.e., all coordinates in the cross product $\mathcal{C}(\mathcal{S}_0) \times \mathcal{C}(\mathcal{S}_1)$ – our technique generates a sequence of pairs Q beginning at the coordinate:

$$(f_L(\mathcal{S}_0), f_U(\mathcal{S}_1))$$

... and ending at the coordinate:

$$(f_U(\mathcal{S}_0), f_L(\mathcal{S}_1))$$

Transitions between intermediate coordinates are determined conditionally:

$$(x_{i+1}, y_{i+1}) = \begin{cases} (x_i + \epsilon, y_i) & \text{if } f(\mathcal{N}|x_i, y_i) < f_T \\ (x_i, y_i - \epsilon) & \text{otherwise} \end{cases}$$

where ϵ is the distance to the next subconfiguration.

In order to guarantee the optimality of our approach, we must ensure that at least one cost-minimizing pair is represented somewhere in this sequence.

Theorem: There exists some coordinate $(x, y) \in Q$ for which $|f(\mathcal{N}|x, y) - f_T|$ is minimal.

Proof: We assume the contrary, and consider some other coordinate $(f(\mathcal{N}_0), f(\mathcal{N}_1))$ that satisfies the property:

$$|f(\mathcal{N}|\mathcal{N}_0, \mathcal{N}_1) - f_T| < |f(\mathcal{N}|x, y) - f_T| \quad \forall (x, y) \in Q$$

For this to be true, one possibility is that $f(\mathcal{N}|\mathcal{N}_0, \mathcal{N}_1) \geq f_T$. Since our algorithm considers all possible values for y, we can define a nonempty set $\mathcal{X}$ that contains all x such that $y = f(\mathcal{N}_1)$ and $(x, y) \in Q$. Our conditional transition ensures that $f(\mathcal{N}|\mathcal{X}_{\max}, \mathcal{N}_1) \geq f_T$. Since $(f(\mathcal{N}_0), f(\mathcal{N}_1)) \notin Q$, we can also infer that $f(\mathcal{N}_0) > \mathcal{X}_{\max}$, and therefore:

$$f(\mathcal{N}|\mathcal{N}_0, \mathcal{N}_1) > f(\mathcal{N}|\mathcal{X}_{\max}, \mathcal{N}_1) \geq f_T$$

implying that $(f(\mathcal{N}_0), f(\mathcal{N}_1))$ cannot be optimal.

A second possibility is that $f(\mathcal{N}|\mathcal{N}_0, \mathcal{N}_1) < f_T$. Since our algorithm considers all possible values for x, we can define a nonempty set $\mathcal{Y}$ that contains all y such that $x = f(\mathcal{N}_0)$ and $(x, y) \in Q$. Our conditional transition ensures that $f(\mathcal{N}|\mathcal{N}_0, \mathcal{Y}_{\min}) < f_T$. Since $(f(\mathcal{N}_0), f(\mathcal{N}_1)) \notin Q$, we can also infer that $f(\mathcal{N}_1) < \mathcal{Y}_{\min}$, and therefore:

$$f(\mathcal{N}|\mathcal{N}_0, \mathcal{N}_1) < f(\mathcal{N}|\mathcal{N}_0, \mathcal{Y}_{\min}) \leq f_T$$

again implying that $(f(\mathcal{N}_0), f(\mathcal{N}_1))$ cannot be optimal.

Since these are the only two possibilities, we conclude (by contradiction) that such a counterexample will not occur. □

References

Abd-El-Barr, M. 2009. Topological network design: a survey. *J. Netw. Comput. Appl.*, 32(3): 501–509.

Achterberg, T. 2009. SCIP: solving constraint integer programs. *Math. Program. Comput.*, 1(1): 1–41.

Alexander, C. K.; and Sadiku, M. N. O. 2016. *Fundamentals of Electric Circuits*. McGraw-Hill Education, 6th edition. ISBN 978-0078028229.

André, J.; Auray, S.; Brac, J.; De Wolf, D.; Maisonnier, G.; Ould-Sidi, M.-M.; and Simonnet, A. 2013. Design and dimensioning of hydrogen transmission pipeline networks. *Eur. J. Oper. Res.*, 229(1): 239–251.

Balakrishnan, A.; Li, G.; and Mirchandani, P. 2017. Optimal network design with end-to-end service requirements. *Operations Research*, 65(3): 729–750.

Barros, M.; Guilherme, J.; and Horta, N. 2010. Analog circuits optimization based on evolutionary computation techniques. *Integr.*, 43(1): 136–155.

Bein, W. W.; Brucker, P.; and Tamir, A. 1985. Minimum cost flow algorithms for series-parallel networks. *Discret. Appl. Math.*, 10(2): 117–124.

Bertsekas, D. P. 1998. *Network Optimization: Continuous and Discrete Models*. Athena Scientific. ISBN 978-1886529021.

Bertsekas, D. P. 2015. *Convex Optimization Algorithms*. Athena Scientific. ISBN 978-1886529281.

Boardman, J. T.; and Meckiff, C. C. 1985. A branch and bound formulation to an electricity distribution planning problem. *IEEE Transactions on Power Apparatus and Systems*, PAS-104(8): 2112–2118.

Boyce, D.; and Soberanes, J. 1979. Solutions to the optimal network design problem with shipments related to transportation cost. *Transportation Research Part B: Methodological*, 13(1): 65–80.

Brylawski, T. 1971. A combinatorial model for series-parallel networks. *Transactions of the American Mathematical Society*, 154: 1–22.

Carlitz, L.; and Riordan, J. 1956. The number of labeled two-terminal series-parallel networks. *Duke Mathematical Journal*, 23(3): 435 – 445.

Cheng, S.-T. 1998. Topological optimization of a reliable communication network. *IEEE Trans. Reliab.*, 47(3): 225–233.

Cheriton, D.; and Tarjan, R. E. 1976. Finding minimum spanning trees. *SIAM Journal on Computing*, 5(4): 724–742.

Davis, M. G.; Smith, E.; Tudor, A.; Sen, K.; Siddiqi, I.; and Iancu, C. 2020. Towards optimal topology aware quantum circuit synthesis. In *Proceedings of the IEEE International Conference on Quantum Computing and Engineering*, 223–234.

Dechter, R. 2003. *Constraint Processing*. Morgan Kaufmann. ISBN 978-1558608900.

Dechter, R.; and Mateescu, R. 2007. AND/OR search spaces for graphical models. *Artif. Intell.*, 171(2-3): 73–106.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artif. Intell.*, 49(1-3): 61–95.

Dionne, R.; and Florian, M. 1979. Exact and approximate algorithms for optimal network design. *Networks*, 9(1): 37–59.

Duffin, R. 1965. Topology of series-parallel networks. *J. Math. Anal. Appl.*, 10(2): 303–318.

Erol, K.; Hendler, J. A.; and Nau, D. S. 1994. HTN planning: complexity and expressivity. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 1123–1128.

Eskandarpour, M.; Dejax, P.; Miemczyk, J.; and Péton, O. 2015. Sustainable supply chain network design: an optimization-oriented review. *Omega*, 54: 11–32.

Fattah, Y. E.; and Dechter, R. 1995. Diagnosing tree-decomposable circuits. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1742–1749.

Foster, R. M. 1932. Geometrical circuits of electrical networks. *Transactions of the American Institute of Electrical Engineers*, 51(2): 309–317.

Fujimoto, K.; Tsuji, Y.; Hirayama, K.; Yasui, T.; Sato, S.; and Kijima, R. 2012. A study on topology optimization of optical circuits consisting of multi-materials. *Journal of Lightwave Technology*, 30(13): 2210–2215.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory & Practice*. Morgan Kaufmann. ISBN 978-1558608566.

Gupta, A.; and Könemann, J. 2011. Approximation algorithms for network design: a survey. *Surveys in Operations Research and Management Science*, 16(1): 3–20.

Hachtel, G. D.; and Somenzi, F. 1996. *Logic Synthesis and Verification Algorithms*. Kluwer Academic Publishers. ISBN 978-0792397465.

Hansen, E. A.; and Zhou, R. 2007. Anytime heuristic search. *J. Artif. Intell. Res.*, 28: 267–297.

Hassin, R.; and Tamir, A. 1986. Efficient algorithms for optimization and selection on series-parallel graphs. *SIAM J. Discret. Math.*, 7: 379–389.

Horowitz, E.; and Sahni, S. 1974. Computing partitions with applications to the knapsack problem. *Journal of the ACM*, 21(2): 277–292.

Hu, J.; and Sapatnekar, S. S. 2001. A survey on multi-net global routing for integrated circuits. *Integr.*, 31(1): 1–49.

IEC 60063. 1952. Series of preferred values and their associated tolerances for resistors and capacitors. Standard, International Electrotechnical Commission.

Itoko, T.; Raymond, R.; Imamichi, T.; and Matsuo, A. 2020. Optimization of quantum circuit mapping using gate transformation and commutation. *Integr.*, 70: 43–50.

Jan, R.-H.; Hwang, F.-J.; and Chen, S.-T. 1993. Topological optimization of a communication network subject to a reliability constraint. *IEEE Trans. Reliab.*, 42(1): 63–70.

Kelareva, E.; Buffet, O.; Huang, J.; and Thiébaux, S. 2007. Factored planning using decomposition trees. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 1942–1947.

Klinz, B.; and Woeginger, G. J. 2004. Minimum-cost dynamic flows: the series-parallel case. *Networks*, 43(3): 153–162.

Korf, R. E. 1998. A complete anytime algorithm for number partitioning. *Artif. Intell.*, 106(2): 181–203.

Korf, R. E. 2009. Multi-way number partitioning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 538–543.

Korf, R. E. 2011. A hybrid recursive multi-way number partitioning algorithm. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 591–596.

Korf, R. E.; Moffitt, M. D.; and Pollack, M. E. 2010. Optimal rectangle packing. *Ann. Oper. Res.*, 179(1): 261–295.

Kurmanbay, A.; Baktybekov, K.; Sakhanov, K.; Syzdykov, A.; and Mukhamediyev, A. 2020. Optimization of series-parallel connection of PV array to mitigate negative influence of partial shading conditions. *IOP Conference Series: Materials Science and Engineering*.

Lavagno, L.; Markov, I.; Scheffer, L.; and Martin, G. 2016. *Electronic Design Automation for Integrated Circuits Handbook*. CRC Press, 2nd edition. ISBN 978-1482254501.

Lengauer, T. 1990. *Combinatorial Algorithms for Integrated Circuit Layout*. Vieweg+Teubner Verlag. ISBN 978-3322921062.

Levitin, G.; Lisnianski, A.; Ben-Haim, H.; and Elmakis, D. 1998. Redundancy optimization for series-parallel multi-state systems. *IEEE Trans. Reliab.*, 47(2): 165–172.

Liu, B.; and Iwamura, K. 2000. Topological optimization models for communication network with multiple reliability goals. *Comput. Math. Appl.*, 39(7): 59–69.

Liu, B.; Wang, Y.; Yu, Z.; Liu, L.; Li, M.; Wang, Z.; Lu, J.; and Fernández, F. V. 2009. Analog circuit optimization system based on hybrid evolutionary algorithms. *Integr.*, 42(2): 137–148.

MacMahon, P. 1892. The combinations of resistances. *The Electrician*, 28: 601–602.

Marinescu, R.; and Dechter, R. 2009. AND/OR branch-and-bound search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17): 1457–1491.

Miao, G.; Himayat, N.; Li, Y. G.; and Swami, A. 2009. Cross-layer optimization for energy-efficient wireless communications: a survey. *Wireless Communications and Mobile Computing*, 9(4): 529–542.

Moffitt, M. D. 2013. Search strategies for optimal multi-way number partitioning. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, 623–629.

Moon, J. W. 1987. Some enumerative results on series-parallel networks. *Ann. Discret. Math.*, 33: 199–226.

Nagurney, A. 2010. Optimal supply chain network design and redesign at minimal total cost and with demand satisfaction. *International Journal of Production Economics*, 128(1): 200–208.

Nam, G.-J.; and Cong, J. 2007. *Modern Circuit Placement: Best Practices and Results*. Springer. ISBN 978-3540719120.

Nau, D. S.; Au, T.; Ilghami, O.; Kuter, U.; Murdock, J. W.; Wu, D.; and Yaman, F. 2003. SHOP2: an HTN planning system. *J. Artif. Intell. Res.*, 20: 379–404.

Neely, M. J. 2010. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1): 1–211.

Nomura, K.; Yamasaki, S.; Yaji, K.; Bo, H.; Takahashi, A.; Kojima, T.; and Fujita, K. 2019. Topology optimization of conductors in electrical circuit. *Structural and Multidisciplinary Optimization*, 59(6): 2205–2225.

Pathak, P. H.; and Dutta, R. 2011. A survey of network design problems and joint design approaches in wireless mesh networks. *IEEE Communications Surveys & Tutorials*, 13(3): 396–428.

Pinedo, M. L. 2016. *Scheduling: Theory, Algorithms, and Systems*. Springer, 5th edition. ISBN 978-3319265780.

Riordan, J.; and Shannon, C. E. 1942. The number of two-terminal series-parallel networks. *Journal of Mathematics and Physics*, 21(1-4): 83–93.

Russell, S.; and Norvig, P. 2020. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, 4th edition. ISBN 978-0134610993.

Saeedi, M.; and Markov, I. L. 2013. Synthesis and optimization of reversible circuits – a survey. *ACM Computing Surveys*, 45(2).

Schreiber, E. L.; Korf, R. E.; and Moffitt, M. D. 2018. Optimal multi-way number partitioning. *Journal of the ACM*, 65(4): 24:1–24:61.

Singh, J.; and Sapatnekar, S. S. 2004. Topology optimization of structured power/ground networks. In *Proceedings of the 2004 International Symposium on Physical Design*, 116–123.

Sloane, N. J. A. 1991. The on-line encyclopedia of integer sequences. https://oeis.org/ (seq. A000110 and A006351).

Takamizawa, K.; Nishizeki, T.; and Saito, N. 1982. Linear-time computability of combinatorial problems on series-parallel graphs. *Journal of the ACM*, 29(3): 623–641.

Tsamardinos, I.; and Pollack, M. E. 2003. Efficient solution techniques for disjunctive temporal reasoning problems. *Artif. Intell.*, 151(1): 43–89.

Valdes, J.; Tarjan, R. E.; and Lawler, E. L. 1982. The recognition of series parallel digraphs. *SIAM Journal on Computing*, 11(2): 298–313.

Yang, S.-T.; and Ephremides, A. 1997. Optimal network design: the base station placement problem. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 3, 2381–2386.

Zhang, L.; and Malik, S. 2002. The quest for efficient boolean satisfiability solvers. In *Proceedings of the 14th International Conference on Computer Aided Verification*, 17–36.

Zheng, F.; Simpson, A. R.; Zecchin, A. C.; and Deuerlein, J. W. 2013. A graph decomposition-based approach for water distribution network optimization. *Water Resources Research*, 49(4): 2093–2109.