5-2007

Connections in Networks: Hardness of Feasibility versus Optimality

Jon Conrad
Cornell University

Carla P. Gomes
Cornell University

Willem-Jan van Hoeve
Cornell University, vanhoeve@andrew.cmu.edu

Ashish Sabharwal
Cornell University

Jordan Suter
Cornell University

Recommended Citation
Proceedings of the Fourth International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR 2007), LNCS 4510, 16- 28.

Connections in Networks: Hardness of Feasibility versus Optimality*

Jon Conrad, Carla P. Gomes,
Willem-Jan van Hoeve, Ashish Sabharwal, and Jordan Suter

Cornell University, Ithaca, NY 14853, USA
{jmc16,jfs24}@cornell.edu; {gomes,vanhoeve,sabhar}@cs.cornell.edu

Abstract. We study the complexity of combinatorial problems that consist of competing infeasibility and optimization components. In particular, we investigate the complexity of the *connection subgraph problem*, which occurs, e.g., in resource environment economics and social networks. We present results on its worst-case hardness and approximability. We then provide a typical-case analysis by means of a detailed computational study. First, we identify an easy-hard-easy pattern, coinciding with the feasibility phase transition of the problem. Second, our experimental results reveal an interesting interplay between feasibility and optimization. They surprisingly show that proving optimality of the solution of the feasible instances can be substantially easier than proving infeasibility of the infeasible instances in a computationally hard region of the problem space. We also observe an intriguing easy-hard-easy profile for the optimization component itself.

1 Introduction

There is a large body of research studying typical-case complexity of decision problems. This work has provided us with a deeper understanding of such problems: we now have a finer characterization of their hardness beyond the standard worst-case notion underlying NP-completeness results, which in turn has led to the design of new algorithmic strategies for combinatorial problems. Nevertheless, while pure decision problems play a prominent role in computer science, most practical combinatorial problems, as they arise in fields like economics, operations research, and engineering, contain a clear optimization objective in addition to a set of feasibility constraints. In our research agenda we are interested in understanding the interplay between feasibility and optimality. We note that there has been some work on the study of the typical-case complexity of pure optimization problems [6, 10], but not concerning problems that naturally combine a feasibility and an optimization component.

As a study case we consider the typical-case complexity of a problem motivated from resource environment economics and social networks, containing

* Research supported by the Intelligent Information Systems Institute (IISI), Cornell University (AFOSR grant F49620-01-1-0076).

competing feasibility and optimization components. Our experimental results show that the complexity profile of this problem introduces several intriguing aspects that do not occur in pure decision problems. A good understanding of these issues will allow researchers to design better algorithms for a range of applications in a variety of domains.

In the context of resource environment economics, our problem is an abstraction of an application that arises in the design of wildlife preserves (see e.g. [1], [3]). In many parts of the world, land development has resulted in a reduction and fragmentation of natural habitat. Wildlife populations living in a fragmented landscape are more vulnerable to local extinction due to stochastic events and are also prone to inbreeding depression. One method for alleviating the negative impact of land fragmentation is the creation of *conservation corridors* (alternatively referred to as wildlife-, habitat-, environmental-, or movement-corridors). Conservation corridors are continuous areas of protected land that link zones of biological significance [9] (see Figure 1). In designing conservation corridors, land use planners generally operate with a limited budget with which to secure the land to make up the corridor. The most environmentally beneficial conservation corridor would entail protecting every piece of land that exists between the areas of biological significance, hereafter referred to as *natural areas* or *reserves*. In most cases, however, purchasing (the development rights to) every piece of available land would be exceedingly expensive for a land trust or government that is operating with a limited budget. The objective is therefore to design corridors that are made up of the land parcels that yield the highest possible level of environmental benefits (the "utility") within the limited budget available.

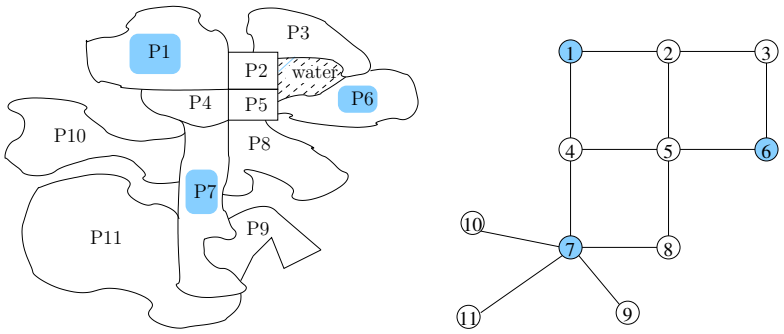


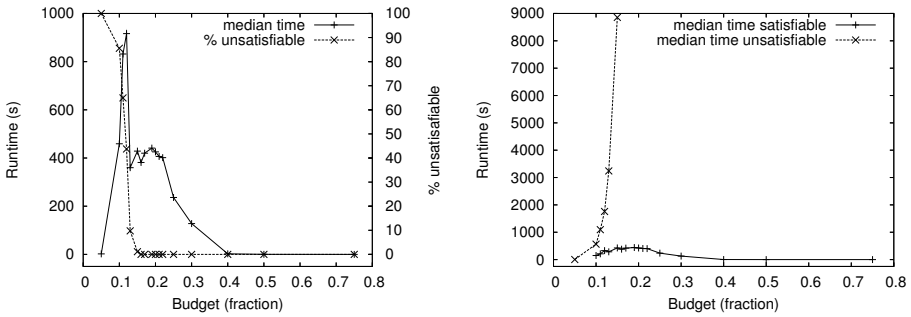
Fig. 1. The "corridor" problem and the corresponding graph representation. The reserves (P1, P6, and P7) and their corresponding vertices are shaded.

In the context of social networks, a similar problem has been investigated by Faloutsos, McCurley, and Tomkins [5]. Here, one is interested, for example, in identifying the few people most likely to have been infected with a disease, or individuals with unexpected ties to any members of a list of other individuals.

This relationship is captured through links in an associated social network graph with people forming the nodes. Faloutsos et al. consider networks containing two special nodes (the "terminals") and explore practically useful utility functions that capture the connection between these two terminal nodes. Our interest, on the other hand, is in studying this problem with the sum-of-weights utility function but with several terminals. In either case, the problem has a bounded-cost aspect that competes with a utility one is trying to maximize.

We formalize the above problems as the *connection subgraph problem*. Somewhat informally, given a graph G on a set of vertices with corresponding utilities, costs, and reserve labels (i.e., whether or not a vertex is a reserve), a set of edges connecting the vertices, and a cost bound (the "budget"), our problem consists of finding a connected subgraph of G that includes all the vertices labeled as reserves and maximizes the total utility, while not exceeding the cost bound. In terms of worst-case complexity, we show that the optimization task associated with the connection subgraph problem is NP-hard, by relating it to the Steiner tree problem. Unlike the original Steiner tree problem, the NP-hardness result here holds even when the problem contains no reserves. We also show that the dual cost minimization problem is NP-hard to approximate within a certain constant factor.

In order to investigate the typical-case complexity of the connection subgraph problem, we perform a series of experiments on semi-structured graphs with randomly placed terminals and randomly generated cost and utility functions. To this end, we introduce a mixed integer linear programming formulation of the problem, which is applied to solve the instances to optimality using Cplex [7]. Figure 2 shows a preview of our results; we defer the details of the experimental setup to Section 5.



a. Hardness and SAT/UNSAT profile

b. Infeasibility vs. optimization

Fig. 2. Aggregated and separated hardness profiles for connection subgraph.

The empirical complexity profile of this problem reveals an interesting interplay between the computational hardness of feasibility testing and optimiza-

tion. In particular, for very low cost bounds (budgets below fraction 0.05 in Figure 2.a), almost all instances are infeasible, which is relatively easy to determine. With increasing cost bounds, one reaches the now standard phase transition point in the feasibility profile, where instances switch from being mostly infeasible to mostly feasible (at roughly budget fraction 0.13 in the plot). At this transition, we see a sharp increase in the complexity of determining feasibility. More interestingly, however, at this phase transition boundary, we have a mixture of feasible and infeasible instances. For the feasible instances, we still need to solve the optimization problem to find (and prove) the maximum utility given the budget constraints. Quite surprisingly, proving such optimality of the solution of these feasible instances can be substantially easier than showing the infeasibility of the other instances in this region (see Figure 2.b). In other words, we have a region in our problem space where the feasible vs. infeasible decision is computationally much harder than proving optimality of the feasible instances. This is surprising because showing optimality also involves a notion of infeasibility: one has to show that there is no solution with a higher utility for the given budget. Intuitively, it appears that the purely combinatorial task of not being able to satisfy the hard constraints of the problem is harder than optimizing solutions in the feasible region.

The second part of the complexity profile of the connection subgraph problem, shown as the lower curve in Figure 2.b, concerns what happens in the feasible region when we further increase the budget beyond the satisfiability phase transition. Almost all instances are now easily shown to be feasible. However, the complexity of finding a solution with the maximum utility and proving its optimality first increases (till budget fraction roughly 0.2 in the plot) and, subsequently, for larger and larger budgets, decreases. Therefore, we have an easy-hard-easy profile in the computational cost of showing optimality, whose peak lies to the right of the feasible to infeasible transition (which, as we saw earlier, is at budget fraction roughly 0.13). In the combined plot of the median runtime of all instances (Figure 2.a), we obtain a curve that peaks around the feasible to infeasible transition because the high cost of proving infeasibility dominates the median cost in the phase transition area.

We note that such easy-hard-easy patterns have been observed in some pure optimization problems before, albeit under a-typical circumstances. For instance, Zhang and Korf [10] identify a similar pattern for the Traveling Salesperson Problem, using a log-normal distribution of the distance function. In our case, the pattern appears to emerge naturally from the model.

These aspects are quite intriguing and require further study. Of course, we do not claim that these observations will hold for all optimization problems that involve a feasibility component. In fact, quite often the feasibility part of optimization tasks is relatively easy provided one has sufficient resources (including budget). However, our study suggests that there may be classes of models or even problems where the feasibility component in and of itself is surprisingly hard, even compared to the optimization aspect. One issue that requires further research is the extent to which the mixed integer programming (MIP) formulation

and the Cplex algorithm are well suited to capture the combinatorial nature of the feasibility problem. In Section 6, we mention two alternative problem formulation/solution methods that might initially appear to be more promising than using Cplex on a pure MIP formulation, but are unlikely to change the overall picture. Lastly, we note that from a practical point of view, this interplay between feasibility and optimization can be quite important. For example, under tight budget constraints, one may want to spend significant computational resources to ensure that no feasible solution exists, before deciding on an increased budget or another relaxation of the problem constraints.

The rest of the paper is organized as follows. In Section 2 we present the connection subgraph problem. We discuss the theoretical complexity of this problem in Section 3. Section 4 describes our Mixed Integer Linear Programming model of the connection subgraph problem. The empirical results are presented in Section 5. Finally, we conclude with a discussion in Section 6.

2 Connection Subgraph Problem

Let $\mathbb{Z}^+$ denote the set $\{0, 1, 2, \ldots\}$ of non-negative integers. The decision version of the connection subgraph problem is defined on an undirected graph as follows:

Definition 1 (Connection Subgraph Problem). *Given an undirected graph $G = (V, E)$ with terminal vertices $T \subseteq V$, vertex costs $c : V \to \mathbb{Z}^+$, vertex utilities $u : V \to \mathbb{Z}^+$, a cost bound $C \in \mathbb{Z}^+$, and a desired utility $U \in \mathbb{Z}^+$, does there exist a vertex-induced subgraph H of G such that*

1. *H is connected,*
2. *$T \subseteq V(H)$, i.e., H contains all terminal vertices,*
3. *$\sum_{v \in V(H)} c(v) \leq C$, i.e., H has cost at most C, and*
4. *$\sum_{v \in V(H)} u(v) \geq U$, i.e., H has utility at least U?*

In this decision problem, we can relax one of the last two conditions to obtain two natural optimization problems:

- **Utility Optimization**: given a cost bound C, maximize the utility of H,
- **Cost Optimization**: given a desired utility U, minimize the cost of H.

3 NP-Completeness and Hardness of Approximation

The connection subgraph problem is a generalized variant of the Steiner tree problem on graphs, with costs on vertices rather than on edges and with utilities in addition to costs. The utilities add a new dimension of hardness to the problem. In fact, while the Steiner tree problem is polynomial time solvable when $|T|$ is any fixed constant [cf. 8], we will show that the connection subgraph problems remains NP-complete even when $|T| = 0$. We prove this by a reduction from the Steiner tree problem. This reduction also applies to planar graphs, for which the Steiner tree problem is still NP-complete [cf. 8].

Theorem 1 (NP-Completeness). *The decision version of the connection subgraph problem, even on planar graphs and without any terminals, is NP-complete.*

Proof. The problem is clearly in NP, because a certificate subgraph H can be easily verified to have the desired properties, namely, connectedness, low enough cost, and high enough utility. For NP-hardness, consider the Steiner tree problem on a graph $\widehat{G} = (\widehat{V}, \widehat{E})$ with terminal set $\widehat{T} \subseteq \widehat{V}$, edge cost function $\widehat{c} : \widehat{E} \to \mathbb{Z}^+$, and cost bound $\widehat{C}$.

An instance of the connection subgraph problem can be constructed from this as follows. Construct a graph $G = (V, E)$ with $V = \widehat{V} \cup \widehat{E}$ and edges defined as follows. For every edge $e = \{v, w\} \in \widehat{E}$, create edges $\{v, e\}, \{w, e\} \in E$. The terminal set remains the same: $T = \widehat{T}$. Overall, $|V| = |\widehat{V}| + |\widehat{E}|$, $|E| = 2|\widehat{E}|$, and $|T| = |\widehat{T}|$. For costs, set $c(v) = 0$ for $v \in \widehat{V}$ and $c(e) = \widehat{c}(e)$. For utilities, set $u(v) = 1$ for $v \in T$ and $u(v) = 0$ for $v \notin T$. Finally, the cost bound for the connection subgraph is $C = \widehat{C}$ and the utility bound is $U = |E|$.

It is easy to verify that the Steiner tree problem on $\widehat{G}$ and $\widehat{T}$ has a solution with cost at most C iff the connection subgraph problem on G and T has a solution with cost at most C and utility at least U. This completes the reduction.

Note that if $\widehat{G}$ is planar, then so is G. Further, the reduction is oblivious to the number of terminals in G. Hence, NP-completeness holds even on planar graphs and without any terminals. □

This immediately implies the following:

Corollary 1 (NP-Hardness of Optimization). *The cost and utility optimization versions of the connection subgraph problem, even on planar graphs and without any terminals, are both NP-hard.*

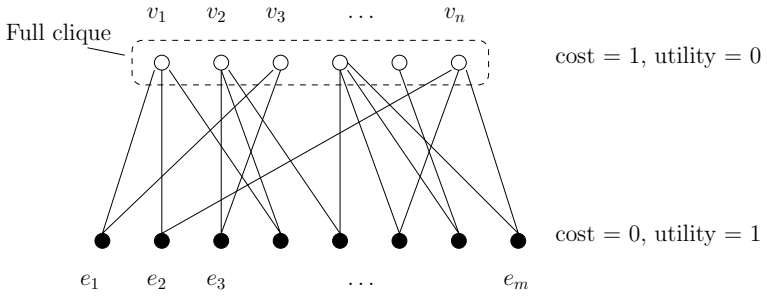
Observe that in the reduction used in the proof of Theorem 1, $\widehat{G}$ has a Steiner tree with cost C' iff G has a connection subgraph with cost C'. Consequently, if the cost optimization version of the connection subgraph instance (i.e., cost minimization) can be approximated within some factor $\alpha \geq 1$ (i.e., if one can find a solution of cost at most α times the optimal), then the original Steiner tree problem can also be approximated within factor α. It is, however, known that there exists a factor α_0 such that the Steiner tree problem *cannot* be approximated within factor α_0, unless P=NP. This immediately gives us a hardness of approximation result for the utility optimization version of the connection subgraph problem. Unfortunately, the best known value of α_0 is roughly $1+10^{-7}$ [cf. 8].

We now describe a different reduction — from the NP-complete Vertex Cover problem — which will enable us to derive as a corollary a much stronger approximation hardness result.

Lemma 1. *There is a polynomial time reduction from Vertex Cover to the connection subgraph problem, even without any terminals, such that the size of the vertex cover in a solution to the former equals the cost of the subgraph in a solution to the latter.*

Proof. We give a reduction along the lines of the one given by Bern and Plassmann [2] for the Steiner tree problem. The reduction is oblivious to the number of terminals, and holds in particular even when there are no terminals.

Recall that a vertex cover of a graph $\widehat{G} = (\widehat{V}, \widehat{E})$ is a set of vertices $V' \subseteq \widehat{V}$ such that for every edge $\{v, w\} \in \widehat{E}$, at least one of v and w is in V'. The vertex cover problem is to determine whether, given $\widehat{G}$ and $C \geq 0$, there exists a vertex cover V' of $\widehat{G}$ with $|V'| \leq C$. We convert this into an instance of the connection subgraph problem. An example of such a graph is depicted in Fig. 3.



Edges in the original graph $\widehat{G}$:
$e_1 = (v_1, v_3), e_2 = (v_1, v_n), e_3 = (v_2, v_3), \ldots, e_m = (v_{n-2}, v_n)$

Fig. 3. Reduction from Vertex Cover

Create a graph $G = (V, E)$ with $V = \widehat{V} \cup \widehat{E}$ and edges defined as follows. For every $v, w \in \widehat{V}, v \neq w$, create edge $\{v, w\} \in E$; for every $e = \{v, w\} \in \widehat{E}$, create edges $\{v, e\}, \{w, e\} \in E$. Overall, G has $|\widehat{V}| + |\widehat{E}|$ vertices and $\binom{\widehat{V}}{2} + 2\widehat{E}$ edges. For costs, set $c(v)$ to be 1 if $v \in \widehat{V}$, and 0 otherwise. For utilities, set $u(e)$ to be 1 if $e \in \widehat{E}$, and 0 otherwise. Finally, fix the set of terminals to be an arbitrary subset of $\widehat{E}$.

We prove that solutions to the connection subgraph problem on G with costs and utilities as above, cost bound C, and desired utility $U = |\widehat{E}|$ are in one-to-one correspondence with vertex covers of $\widehat{G}$ of size at most C.

First, let vertex-induced subgraph H of G be a solution to the connection subgraph instance. Let $V' = V(H) \cap \widehat{V}$. We claim that V' is a vertex cover of $\widehat{G}$ of size at most C. Clearly, $|V'| \leq C$ because of the cost constraint on H. To see that V' is indeed a vertex cover of $\widehat{G}$, note that (A) because of the utility constraint, V' must contain *all* of the vertices from $\widehat{E}$, and (B) because of the connectedness constraint, every such vertex must have at least one edge in $E(H)$, i.e., for each $e = \{v, w\} \in \widehat{E}$, V' must include at least one of v and w.

Conversely, let V' be a vertex cover of $\widehat{G}$ with at most C vertices. This directly yields a solution H of the connection subgraph problem: let H be the

subgraph of G induced by vertices $V' \cup \widehat{E}$. By construction, H has the same cost as V' (in particular, at most C) and has utility exactly U. Since V' is a vertex cover, for every edge $e = \{v, w\} \in \widehat{E}$, at least one of v and w must be in V', which implies that H must have at least one edge involving e and a vertex in V'. From this, and the fact that all vertices of V' already form a clique in H, it follows that H itself is connected.

This settles our claim that solutions to the two problem instances are in one-to-one correspondence, and finishes the proof. □

Combining Lemma 1 with the fact that the vertex cover problem is NP-hard to approximate within a factor of 1.36 [4] immediately gives us the following:

Theorem 2 (APX-Hardness of Cost Optimization). *The cost optimization version of the connection subgraph problem, even without any terminals, is NP-hard to approximate within a factor of* 1.36*.*

4 Mixed Integer Linear Programming Model

Next we present the Mixed Integer Linear Programming Model (MIP model) for the connection subgraph problem, that we used in our experiments. Let $G = (V, E)$ be the graph under consideration, with $V = \{1, \ldots, n\}$.

For each vertex $i \in V$, we introduce a binary variable x_i, representing whether or not i is in the connected subgraph. Then, the objective function and budget constraint are stated as:

$$\text{maximize} \quad \sum_{i \in V} u_i x_i, \tag{1}$$

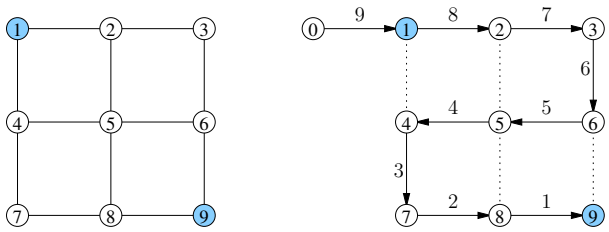
$$\text{s.t.} \sum_{i \in V} c_i x_i \leq C, \tag{2}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{3}$$

To ensure the connectivity of the subgraph, we apply a particular network flow model, where the network is obtained by replacing all undirected edges $\{i, j\} \in E$ by two directed edge $\{i, j\}$ and $\{j, i\}$. First, we introduce a source vertex 0, with maximum total outgoing flow n. We arbitrarily choose one terminal vertex $t \in T$, and define a directed edge $\{0, t\}$ to insert the flow into the network, assuming that there exists at least one such vertex.[1] Then, by demanding that the flow reaches all terminal vertices, the edges carrying flow (together with the corresponding vertices) represent a connected subgraph. To this end, each of the vertices with a positive incoming flow will act as a 'sink', by 'consuming' one unit of flow. In addition, flow conservation holds: for every vertex the amount of incoming flow equals the amount of outgoing flow.

More formally, for each edge $\{i, j\} \in E$, we introduce a nonnegative variable y_{ij} to indicate the amount of flow from i to j. For the source, we introduce a

[1] If there are no terminal vertices specified, we add edges from the source to all vertices in the graph, and demand that at most one of these edges is used to carry flow.



a. Original graph b. Feasible flow

Fig. 4. Flow representation of the connection subgraph problem on a graph with 9 vertices. The terminal vertices 1 and 9 are shaded.

variable $x_0 \in [0, n]$, representing the eventual residual flow. The insertion of the flow into the network is then stated as:

$$x_0 + y_{0t} = n, \tag{4}$$

where $t \in T$ is arbitrarily chosen. Each of the vertices with positive incoming flow retains one unit of flow, i.e., $(y_{ij} > 0) \Rightarrow (x_j = 1), \forall \{i, j\} \in E$. We convert this relation into a linear constraint:

$$y_{ij} < n x_j, \ \forall \{i, j\} \in E. \tag{5}$$

The flow conservation is modeled as:

$$\sum_{i:\{i,j\}\in E} y_{ij} = x_j + \sum_{i:\{j,i\}\in E} y_{ij}, \ \forall j \in V. \tag{6}$$

Finally, terminal vertices retain one unit of flow:

$$x_t = 1, \ \forall t \in T. \tag{7}$$

In Figure 4 we give an example of our flow representation, where we omit the costs for clarity. Figure 4.a presents a graph on 9 vertices with terminal vertices 1 and 9. In Figure 4.b, a feasible flow for this graph is depicted, originating from the source 0, with value 9. It visits all vertices, while each visited vertex consumes one unit of flow. The thus connected subgraph contains all vertices in this case, including all terminal vertices.

5 Computational Hardness Profiles

We next perform a detailed empirical study of the connection subgraph problem. In this study, our parameter is the feasibility component of the problem, i.e., the cost bound (or budget). For a varying budget, we investigate the satisfiability of the problem, as well as its computational hardness with respect to proving infeasibility or optimality.

In our experiments, we make use of semi-structured graphs, with uniform random utility and cost functions. The graphs are composed of an $m \times m$ rectangular lattice or grid, where the order m is either 6, 8, or 10. This lattice graph is motivated by the structure of the original conservation corridors problem. In this lattice, we place k terminal vertices, where k is 0, 3, 10, or 20. When $k \geq 2$, we place two terminal vertices in the 'upper left' and 'lower right' corners of the lattice, so as to maximize the distance between them and "cover" most of the graph. This is done to avoid the occurrence of too many pathological cases, where most of the graph does not play any role in constructing an optimal connection subgraph. The remaining $k - 2$ terminal vertices are placed uniformly at random in the graph. To define the utility and cost functions, we assign uniformly at random a utility and a cost from the set $\{1, 2, \ldots, 10\}$ to each vertex in the graph. The cost and utility functions are uncorrelated.

In the figures below, each data point is based on 100 random instances or more, at a given budget. For the figures comparing infeasible and feasible instances, this means that the sum of the feasible and infeasible instances at each budget is at least 100. The hardness curves are represented by median running times over all instances per data point, while for the feasibility curves we take the average. As the scale for the budget (on the x-axis), we use the following procedure. For every instance, we compute the total cost of all vertices. The budget is calculated as a fraction of this total cost. We plot this fraction on the x-axis. All our experiments were conducted on a 3.8 GHz Intel Xeon machine with 2 GB memory running Linux 2.6.9-22.ELsmp. We used Cplex 10.1 [7] to solve the MIP problems.

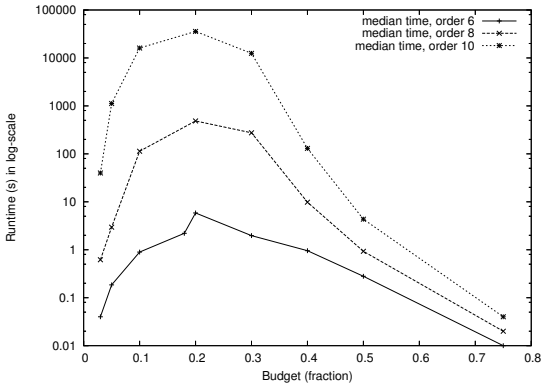
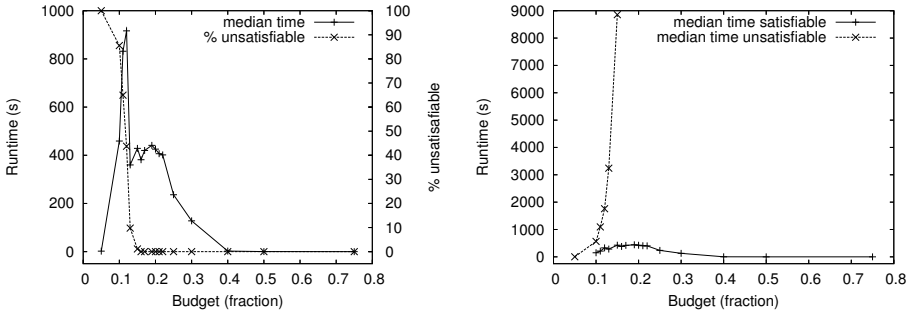


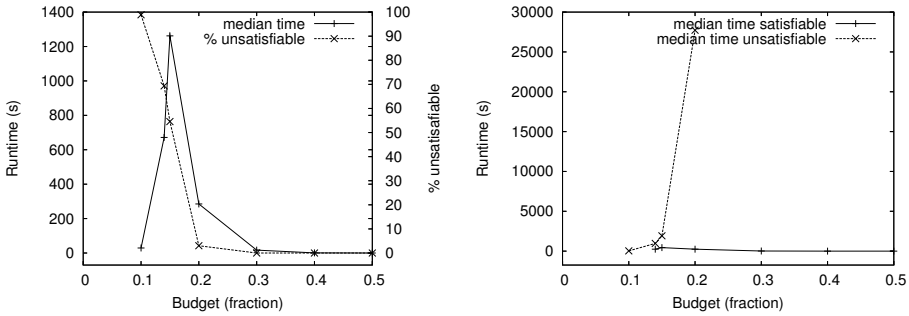
Fig. 5. Hardness profile for lattices of order 6, 8, and 10, without terminal vertices.

First, we present computational results on graphs without terminal vertices. These problems are always satisfiable, and can thus be seen as pure optimization problems. Figure 5 shows the hardness profile (i.e., the running time) on lattices



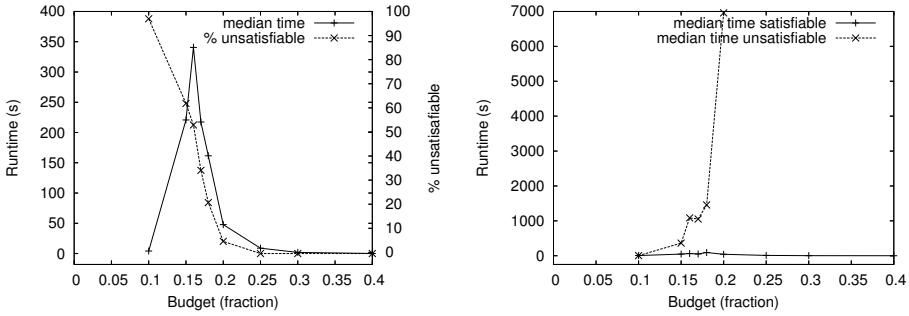
a. Hardness and SAT/UNSAT profile b. Infeasibility vs. optimization

Fig. 6. Hardness and satisfiability profiles for lattices of order 10 with 3 terminals.



a. Hardness and SAT/UNSAT profile b. Infeasibility vs. optimization

Fig. 7. Hardness and satisfiability profiles for lattices of order 10 with 10 terminals.



a. Hardness and SAT/UNSAT profile b. Infeasibility vs. optimization

Fig. 8. Hardness and satisfiability profiles for lattices of order 10 with 20 terminals.

of order 6, 8, and 10. Notice that the median time is plotted in log-scale in this figure. The plots clearly indicate an easy-hard-easy pattern for these instances, even though they are all feasible with respect to the budget. As remarked earlier, such patterns have been observed earlier in some pure optimization problems, but only under specific random distributions.

Second, we turn our attention to graphs with terminal vertices. In Figure 6.a, we show the hardness profile of lattices of order 10, with 3 terminals. In addition, the satisfiability profile is shown in this figure: we plot the percentage of unsatisfiable instances with respect to a varying budget. Figures 7.a and 8.a present similar graphs for lattices of order 10, with 10 and 20 terminals, respectively. In all figures, we see a sharp phase transition from a region in which almost all instances are unsatisfiable, to a region in which almost all instances are satisfiable (when the budget fraction is around 0.15). Furthermore, again these problems exhibit an easy-hard-easy pattern, the peak of which coincides with the satisfiability phase transition with respect to the budget. Similar relations between the peak of computational hardness and feasibility phase transitions have been demonstrated often before for pure satisfiability problems. However, we are unaware of such results for problems combining both a feasibility and an optimality aspect.

Our experiments also indicate an easy-hard-easy pattern for the hardness of the problem, depending number of terminals in the graph. For 10 terminals, the problems are considerably more difficult than for 3 or 20 terminals. Intuitively, this can be explained by two rivaling aspects: the difficulty of connecting k terminals, and the complexity on $n - k$ free variables. As k increases, it is more difficult to connect the terminals. However, when k is large, the resulting problem on $n - k$ variables becomes easy.

Finally, we compare the hardness of optimization to the hardness of proving infeasibility. To this end, we separate the hardness profiles for satisfiable and unsatisfiable problem instances. The resulting plots for lattices of order 10 with 3, 10, and 20 terminals are depicted in Figure 6.b, Figure 7.b, and Figure 8.b, respectively. In these figures, the curve for unsatisfiable instances represents the hardness of proving infeasibility, while the curve for satisfiable instances represents the hardness of proving optimality. Clearly, proving infeasibility becomes increasingly more difficult when the budget increases, especially inside the phase transition region. At the same time, the difficulty of proving optimality does not exhibit this extreme behavior. In fact, when the budget fraction is around 0.15, we observe that proving infeasibility takes up to 150 times longer than proving optimality.

6 Summary and Discussion

In this work, we investigated the interplay between the computational tasks of feasibility testing and optimization. We studied in detail the connection subgraph problem, for which we presented theoretical worst-case complexity results, as well as empirical typical-case results. Our experiments reveal interesting trade-offs

between feasibility testing and optimization. One of our main observations is that proving infeasibility can be considerably more difficult than proving optimality in a computationally hard region of the problem space. In addition to this, we identified a satisfiability phase transition coinciding with the complexity peak of the problem. Somewhat more surprisingly, for the optimization component itself, we discovered an easy-hard-easy pattern based on the feasibility parameter, even when the underlying problems are always satisfiable.

In our experimental results, we have applied a mixed integer linear programming model in conjunction with the solver Cplex. Naturally, one could argue that a different solver or even a different model could have produced different results. For example, one might propose to check separately the feasibility of the cost constraint before applying a complete solver. Indeed, checking feasibility of the cost constraint is equivalent to the metric Steiner tree problem. Although this latter problem is solvable in polynomial time for a constant number of terminals, it is likely not to be fixed parameter tractable [8]. Hence, it appears unrealistic to apply such a separate feasibility check as a pre-processor before using a complete solution technique.

Another direction is to apply a constraint programming (CP) model, which could perhaps better tackle the feasibility aspect of the problem. However, a good CP model should ideally capture the cost constraint as a whole, for example as a global constraint. For the same reason as above, it is unlikely that an efficient and effective filtering algorithm exists for such a constraint. Moreover, a CP model by itself is not particularly suitable for the optimization component. More specifically, for the connection subgraph problem the objective is a weighted sum, which is known to be difficult to handle by constraint solvers. Nevertheless, a hybrid constraint programming and mixed integer programming approach might be effective for this problem, which we leave open as future work.

References

[1] A. Ando, J. Camm, S. Polasky, and A. Solow. Special distributions, land values, and efficient conservation. *Science*, 279(5359):2126–2128, 1998.

[2] M. W. Bern and P. E. Plassmann. The Steiner tree problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

[3] J. D. Camm, S. K. Norman, S. Polasky, and A. R. Solow. Nature reserve site selection to maximize expected species covered. *Operations Research*, 50(6):946–955, 2002.

[4] I. Dinur and S. Safra. On the hardness of approximating minimum vertex cover. *Annals of Mathematics*, 162(1):439–486, 2005.

[5] C. Faloutsos, K. McCurley, and A. Tomkins. Fast discovery of connection subgraphs. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 118–127. ACM Press, 2004.

[6] I. Gent and T. Walsh. The TSP Phase Transition. *Artificial Intelligence*, 88(1–2): 349–358, 1996.

[7] ILOG, SA. CPLEX 10.1 Reference Manual, 2006.

[8] H. J. Prömel and A. Steger. *The Steiner Tree Problem: A Tour Through Graphs, Algorithms, and Complexity.* Vieweg, 2002.

[9] D. Simberloff, J. Farr, J. Cox, and D. Mehlman. Movement corridors: Conservation bargains or poor invesments? *Conservation Biology*, 6:493–504, 1997.

[10] W. Zhang and R. Korf. A Study of Complexity Transitions on the Asymmetric Traveling Salesman Problem. *Artificial Intelligence*, 81:223–239, 1996.