An Incremental Model for Combinatorial Minimization

Jeff Hartline Alexa Sharp

Cornell University, Ithaca, NY 14853
{jhartlin, asharp}@cs.cornell.edu

Abstract

Traditional optimization algorithms are concerned with static input, static constraints, and attempt to produce static output of optimal value. Recent literature has strayed from this conventional approach to deal with more realistic situations in which the input changes over time. *Incremental optimization* is a new framework for handling this type of dynamic behavior. We consider a general model for producing incremental versions of traditional covering problems along with several natural incremental metrics. Using this model, we demonstrate how to convert conventional algorithms into incremental algorithms with only a constant factor loss in approximation power. We introduce incremental versions of min cut, edge cover, and (k, r)-center and present some hardness results. Lastly, we discuss how the incremental model can help us more fully understand online problems and their corresponding algorithms.

1 Introduction

Suppose a cell phone company wants to provide service to a growing community. They do so by building cell towers, which serve all clients within a fixed radius. Initially the customer base is small, and thus few towers suffice to satisfy the demand for coverage. As demand increases, more towers are necessary. Given a projection for how demand will grow annually, the company would like to know where to build towers each year so that all customers are provided for at any given time. How should the company plan its building so that the number of towers built over the first ℓ years is not much more than the optimal number of towers needed to satisfy just the year ℓ demand?

This problem is reminiscent of (k, r)-center [1], which takes as input a set of client locations in a metric space and determines if k centers can be selected from the client set so that every client is within a radius r of some center. Conventional algorithms for (k, r)-center can be applied to our cell tower example for any static customer base, but they cannot handle the time-dependencies intrinsic to this type of real-world scenario. Instead, we need an *incremental* version of (k, r)-center. The client set grows at discrete time intervals and there is no fixed parameter k; this value must change as new centers open to accommodate all current clients. As it is impractical to relocate towers once they are constructed, the set of open centers increases monotonically; once built, a center remains part of the solution in perpetuity. If not for this incremental constraint, we would simply use the optimal set of towers at each time step. Thus, one goal is to compare the number of centers used in each step to the number required by the current client base. We want to minimize the maximum of this ratio over all time steps. An algorithm is said to be α-*competitive* if this maximum is no more than α. Alternatively, if we are more interested in overall performance, our goal might be to minimize the sum of the solution costs over all time steps. If each tower has an annual operating cost, minimizing net cost over all years corresponds to minimizing this sum.

Most combinatorial minimization problems can be extended to incremental optimization problems. The *incremental cut* problem is defined on a network with source s, sink t, and a growing set of edges. A solution is a sequence of cuts, one per time step, such that each cut disconnects s from t given the current edge set and earlier cuts are contained in all subsequent cuts. In *incremental edge cover*, we are given an edge-weighted graph and a non-decreasing sequence of target vertex sets. We want an edge cover for each target set such that each cover builds on all prior covers.

Motivation and Related Work. Recent interest in incremental optimization has focused predominantly on cardinality constrained minimization problems. Such problems specify a fixed parameter k that designates the cardinality of feasible solutions. Incremental versions of these problems require solutions for all values of k, such that the kth solution contains all prior solutions. Mettu and Plaxton [2] study incremental uncapacitated k-median and give a 29.86-competitive algorithm. Plaxton [3] introduces incremental facility location and gives a $(1+\epsilon)\alpha$-competitive algorithm, providing that an α-approximation for uncapacitated facility location exists; this results in a 12.16-competitive algorithm. Gonzales [4] gives a 2-approximation algorithm for k-center, which is also a 2-competitive algorithm for the incremental k-center problem studied by [2, 3, 5, 6]. Lin et al.[5] present a general framework for cardinality constrained minimization problems, resulting in approximation algorithms for incremental k-vertex cover and k-set cover, an improved approximation algorithm for incremental k-median, and alternative approximation algorithms for incremental k-MST and incremental facility location.

The cell tower example, however, motivates an alternative approach to incremental optimization that is the focus of this paper. Specifically, the incremental aspect of the tower placement problem is driven by a growing customer base demanding service, as opposed to a parameter increasing the number of towers to build. Thus rather than require solutions to be of a specific cardinality, we require the set of solutions to be *monotonic*, i.e. supersets of feasible solutions are feasible, or in the case of maximization problems, subsets of feasible solutions are feasible. Hartline and Sharp [7] introduce incremental versions of max flow, a monotonic maximization problem, and in [8] they formalize a general model and algorithmic framework for monotonic packing problems, achieving an $O(\frac{\alpha}{\log k})$-approximation for any k-level incremental packing problem with an α-approximation algorithm. Although better algorithms exist for incremental bipartite matching and incremental knapsack, the algorithm is tight in the case of max flow. Our paper will address analogous issues with respect to combinatorial covering problems, a large class of monotonic problems including classic versions of min cut and vertex cover.

Online problems share many similarities with the incremental model. For both types of problems, input arrives over time and solutions build incrementally. The main difference is that online algorithms act with no knowledge of future input [9, 10] whereas incremental algorithms have full knowledge of the input sequence; this difference turns out to be crucial. The performance of an online algorithm, as measured by its *competitive ratio* [11], is influenced by two principle factors. The first factor is that online solutions are constrained to keep all intermediate solutions, and it may not be possible to obtain reasonable solutions at each level while simultaneously guaranteeing a final solution with good competitive ratio. We call this the *incremental constraint*. The second factor is that online algorithms have no knowledge of future input, and thus an adversary may select input sequences that prevent any one algorithm from obtaining a good competitive ratio. We call this the *adversarial constraint*. Both of these factors may contribute to an algorithm's poor performance, but the competitive ratio fails to discern which is the more significant. In contrast, incremental problems are only burdened by the incremental constraint, and thus can be viewed as the offline version of online problems. We can therefore use an incremental problem's approximation results to determine which constraint contributes most to the hardness of an online problem. If the performance gap between an online and incremental problem is large then we conclude the adversarial constraint is the dominating factor. In this case, resources should not be wasted investigating improved online algorithms, but instead should be spent developing

better predictions of future input to counteract the adversarial constraint. On the other hand, if the performance gap is small then we conclude the adversarial constraint does not contribute significantly to the hardness of the problem, but rather the incremental constraint is at fault. Online algorithms have been studied in many contexts, including bin packing[12], graph coloring [13], bipartite matching [14], and monotone set systems [15]. The relationship between online and incremental problems is illustrated in Section 5.

Stochastic optimization also resembles our incremental framework in that instances have multi-stage input and incremental solutions. However, the input is probabilistic, costs are stage-dependent, and the goal is to find a final solution of optimal expected cost. We motivate our general models by those developed for stochastic problems [16, 17, 18]. General models for single-level optimization problems are available in [19, 20].

Our Results. We modify the general incremental model of [8] and analyze the complexity of incremental cut and incremental edge cover with respect to three different objective functions: minimum ratio, minimum sum, and demand. We find that incremental edge cover remains in P in many cases, whereas incremental cut is NP-hard. Although this is analogous to results in [8] for incremental matching and incremental flow, the methods of proof are fundamentally different. We also introduce incremental r-domination, a monotonic version of (k, r)-center [1].

Our central contribution is a general technique to translate exact or approximate algorithms for monotonic covering problems into approximation algorithms for their corresponding incremental versions. In particular, given an α-approximation for a covering problem, we present a 4α-approximation for its incremental version for both the min sum and min ratio objectives. This scheme is more efficient and more general than the $O(\frac{\alpha}{\log k})$ approximation given for incremental packing problems in [8], which only works for the max sum metric.

After applying our approximation scheme to min cut, edge cover, and r-domination, we use it to investigate the complexity of online Steiner tree. Interestingly, the 4α-approximation reveals a gap between the online and incremental versions of Steiner tree, demonstrating that the hardness of online Steiner tree is due primarily to adversarial constraints.

Section 2 introduces notation for our general model of incremental minimization. Section 3 demonstrates how min cut, edge cover and (k, r)-center can be cast into this model, and provides accompanying complexity results for these problems. We present our general approximation techniques in Section 4 and discuss its implications to online Steiner tree in Section 5. Finally, Section 6 concludes by enumerating several potential extensions for our incremental model.

2 The General Model

Our model for combinatorial minimization problems is taken with slight modification from the general model for maximization problems in [8], which adapts the models of [16, 17]. As we will demonstrate, this slight modification has significant impact on results, and thus warrants the separate treatment provided by this paper. For completeness, we now present the model.

Single-Level Problems. We represent a single-level abstract optimization problem Π as a tuple $(X, \mathcal{F}, v)$. The set X is a collection of objects, and feasible solutions are subsets of X that collectively achieve a certain goal [21]. The set of all such solutions is $\mathcal{F} \subseteq 2^X$, and $v : \mathcal{F} \to \mathbb{R}$ is a cost function on these solutions. For a large class of problems, including most packing and covering problems, we associate a weight w_x with every object x of X and define $v(S) = \Sigma_{x \in S}\ w_x$. An optimal solution $OPT(X, \mathcal{F}, v)$, or $OPT(\mathcal{F})$ if X and v are understood, is a subset $S \subseteq X,\ S \in \mathcal{F}$ optimizing $v(S)$.

This paper focuses on covering problems, a class of minimization problems exhibiting *monotonicity*: $S \in \mathcal{F}$ and $S' \supseteq S$ imply $S' \in \mathcal{F}$. In other words, any superset of a feasible solution is also feasible. Min cut, edge cover, vertex cover, and many other classic minimization problems are all monotonic. See [8] for analogous treatment of monotonic packing problems.

Incremental Problems. We extend the notion of an abstract optimization problem to a multi-level incremental optimization problem. The k-level *incremental version* of Π, denoted Π^k, consists of k instances of Π with the same object set X and cost function v, but each with its own feasible set. Thus an instance of Π^k is represented as a tuple $(X, (\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k), v)$. A solution is a tuple $\mathbf{S} = (S_1, S_2, \ldots, S_k)$ that meets the *feasibility constraint* $S_\ell \in \mathcal{F}_\ell$ and the *incremental constraint* $S_\ell \subseteq S_{\ell+1}$. We further assume $\mathcal{F}_1 \supseteq \mathcal{F}_2 \supseteq \ldots \supseteq \mathcal{F}_k$, a reasonable assumption for monotonic covering problems.

There are three objective functions for incremental covering problems, dual to the objectives introduced in [8] for incremental packing problems. The *minimum ratio* objective is to obtain the smallest possible ratio of each level's optimal solution: find $\mathbf{S}$ minimizing $\mathcal{R}(\mathbf{S}) = max_\ell \frac{v(S_\ell)}{v(OPT(X,\mathcal{F}_\ell,v))}$. This is the same as the competitive ratio for online algorithms, and is a standard metric for incremental problems [3, 2]. If we are more concerned with overall performance, as opposed to fairness between levels, the ratio objective might not be appropriate. The *minimum sum* objective is to minimize the sum of the solutions over all levels: find $\mathbf{S}$ minimizing $\mathcal{V}(\mathbf{S}) = \Sigma_\ell v(S_\ell)$. Lastly, the *demand* objective takes demands $d_1, \ldots, d_k$ and seeks an incremental solution with $v(S_\ell) \leq d_\ell$.

We now demonstrate how min cut, edge cover, and (k, r)-center all fall into this framework.

3 Three Covering Problems

3.1 Incremental Cut

The *minimum cut* problem is defined on a graph $G = (V, E)$ with source s, sink t, and edge weights w_e; the objects are the edges of G, and feasible solutions are subsets of E that cut s from t. The cost of a cut S is the sum of the weights of the edges it contains: $v(S) = \Sigma_{e \in S} w_e$. *Incremental cut* (IC) is defined on a similar graph, only the edges E appear over k discrete time intervals $E_1 \subseteq E_2 \subseteq \ldots \subseteq E_k = E$. A solution is a sequence of s-t cuts $(S_1, S_2, \ldots, S_k)$ such that S_ℓ is an s-t cut of (G, E_ℓ) and $S_\ell \subseteq S_{\ell+1}$. G may be directed or undirected in either of the above problems.

It is well-established that directed and undirected min cut have polynomial-time algorithms. In contrast, directed and undirected IC are intractable, as stated by Theorems 3.1-3.2. Their proofs, via non-trivial reduction, can be found in Appendix A.

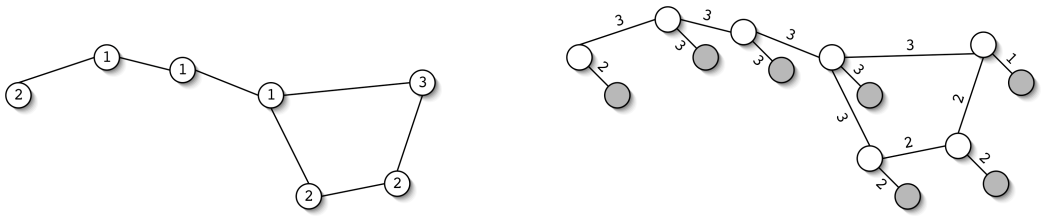
Theorem 3.1 *Min ratio, min sum, and demand IC are NP-hard for directed unweighted graphs, $k \geq 2$.*

Theorem 3.2 *Demand IC is NP-hard for undirected unweighted graphs, $k \geq 2$.*

Theorems 3.1-3.2 are of interest as they demonstrate how the addition of the incremental aspect to a problem can increase its complexity significantly. This phenomenon was first observed for max flow [7], another polynomial-time problem that becomes intractable under the incremental framework.

3.2 Incremental Edge Cover

The *edge cover* problem is defined on an undirected graph $G = (V, E)$ with edge weights w_e; the objects are the edges of G, and feasible solutions are edge sets that cover all vertices. The cost of an edge cover S is $v(S) = \Sigma_{e \in S} w_e$. Strict application of our incremental model to edge cover has a negligible effect on the nature of the problem. We instead consider *subset edge cover*, a generalization of edge cover defined



(a) (b)

Figure 1: 1(a): A three-level instance of IEC. Labels indicate what level vertices first appear in the target set. 1(b): A single-level subset edge cover instance constructed from the graph in 1(a).

in [22]. Along with G and w there is an additional input $T \subseteq V$, where T is the target set that must be covered. Subset edge cover can be solved in polynomial time by reducing it to an instance of edge cover [22]. We define *incremental edge cover* (IEC) on an undirected graph $G = (V, E)$ with an increasing sequence of k target sets $T_1 \subseteq T_2 \subseteq \ldots \subseteq T_k$. A solution is a sequence of edge covers $(S_1, S_2, \ldots, S_k)$ such that S_ℓ is a subset edge cover of T_ℓ in G, and $S_\ell \subseteq S_{\ell+1}$.

Whereas min ratio IEC is NP-hard, min sum IEC is tractable. We provide a polynomial-time algorithm for min sum IEC but refer to Appendix B for its proof of correctness. The hardness proof for min ratio IEC, which uses a reduction from partition, is also in Appendix B.

Theorem 3.3 *Min ratio IEC is NP-hard for 2 levels.*

Theorem 3.4 *Min sum IEC is in P.*

IEC Preliminaries. Given an IEC instance $(G, w, \mathbf{T} = T_1, T_2, \ldots, T_k)$, we call vertex $t \in T_k$ a *level ℓ vertex* if t first appears in target set T_ℓ. We denote the level of t as ℓ_t. For each such vertex t, let e_t denote the min cost edge incident to t. We say an incremental solution is *reduced* if for each edge $\{u, v\}$ either (1) $u, v \in T_k$ and $\{u, v\}$ is added to our solution at level $\min\{\ell_u, \ell_v\}$, or (2) $\{u, v\} = e_t$ for some $t \in T_k$ and is added at level ℓ_t. The algorithm follows, with illustration in Figure 1.

IEC Algorithm. We construct a single-level subset edge cover instance (G', w', T_k) such that covers of G' correspond to reduced incremental covers of G of equal cost. The construction occurs in two phases:

(i) Begin with the subgraph of G induced by T_k. For each $e = \{u, v\}$ set $w'(e) = (k+1-min\{\ell_u, \ell_v\}) \cdot w(e)$: the cost of using e to cover u and v in a reduced cover.

(ii) For each $u \in T_k$, create a new vertex $\hat{u}$ and edge $\{u, \hat{u}\}$ of weight $(k + 1 - \ell_u) \cdot w(e_u)$: the cost of using e_u to cover only u in a reduced cover.[1]

We solve this single-level instance to obtain a min-cost subset edge cover S, and build incremental solution $\mathbf{S} = (S_1, S_2, \ldots, S_k)$ as follows: if $\{u, v\} \in S$ then place $\{u, v\}$ in $S_{\min\{\ell_u, \ell_v\}}$, and if $\{u, \hat{u}\} \in S$ then place e_u in S_{ℓ_u}. Return $\mathbf{S}$ as an optimal incremental cover.

3.3 Incremental (k, r)-Center

As a final example, recall the cell phone tower scenario from Section 1, and the (k, r)-center problem that models it for any static input. There are two natural ways to approach (k, r)-center:

[1]Not all of these edges are necessary, but they do not increase the size of G' excessively.

(i) *k-center*: fix the number of centers k and minimize the maximum distance between any client and its closest center. Gonzalez [4] gives a 2-approximation for k-center.

(ii) *r-domination* [1, 23]: fix the radius r and cover all clients with as few centers as possible. The current best algorithm for r-domination is a general $O(\log n)$-approximation for set cover [1].

Incremental k-center has been widely studied [2, 3, 5, 6]. The input is the same as for k-center, but the goal is to find a sequence of k centers such that, for any ℓ, the first ℓ centers are competitive with the optimal ℓ-center solution. Gonzalez' 2-approximation [4] is online, and thus applies to the incremental case. However, k-center is not a monotonic covering problem and therefore our model does not apply.

In contrast, there is no known version of incremental r-domination but we can apply our general model. The objects are potential center locations and feasible solutions are sets of centers that cover an incrementally increasing client base. This precisely models the cell tower application. Unfortunately, incremental r-domination is NP-hard, as it contains the conventional r-domination problem as a special case. This fact, along with the hardness of incremental cut and min ratio incremental edge cover, motivates the need for a general approximation scheme for incremental problems. We give two such results in the following section.

4 General Approximation Algorithms

We use the general model of Section 2 to convert single-level algorithms into incremental algorithms for the min sum and min ratio metrics. Theorem 4.1 considers 2-level incremental instances, whereas Theorem 4.2 deals with an arbitrary number of levels.

Theorem 4.1 *If algorithm ALG α-approximates Π, we can $(\varphi \cdot \alpha)$-approximate min sum and min ratio versions of Π^2, where φ is the golden ratio*[2]*.*

Proof. Run ALG on both single-level input to obtain $v(ALG(\mathcal{F}_1)) \leq \alpha \cdot v(OPT(\mathcal{F}_1))$ and $v(ALG(\mathcal{F}_2)) \leq \alpha \cdot v(OPT(\mathcal{F}_2))$. We define incremental solution (S_1, S_2) as follows:

(i) If $v(ALG(\mathcal{F}_2)) > \varphi \cdot v(ALG(\mathcal{F}_1))$, then $S_1 = ALG(\mathcal{F}_1)$, $S_2 = S_1 \cup ALG(\mathcal{F}_2)$, so that

$$\begin{array}{rcl@{\qquad}rcl} v(S_1) & = & v(ALG(\mathcal{F}_1)) & v(S_2) & \leq & v(ALG(\mathcal{F}_1)) + v(ALG(\mathcal{F}_2)) \\ & \leq & \alpha \cdot v(OPT(\mathcal{F}_1)) & & \leq & \frac{1}{\varphi} \cdot v(ALG(\mathcal{F}_2)) + v(ALG(\mathcal{F}_2)) \\ & & & & \leq & (1 + \frac{1}{\varphi}) \cdot \alpha \cdot v(OPT(\mathcal{F}_2)). \end{array}$$

(ii) If $v(ALG(\mathcal{F}_2)) \leq \varphi \cdot v(ALG(\mathcal{F}_1))$, then $S_1 = ALG(\mathcal{F}_2) = S_2$, so that

$$\begin{array}{rcl@{\qquad}rcl} v(S_1) & = & v(ALG(\mathcal{F}_2)) & v(S_2) & = & v(ALG(\mathcal{F}_2)) \\ & \leq & \varphi \cdot v(ALG(\mathcal{F}_1)) & & \leq & \alpha \cdot v(OPT(\mathcal{F}_2)) \\ & \leq & \varphi \cdot \alpha \cdot v(OPT(\mathcal{F}_1)). & & & \end{array}$$

The optimal sum is at least $v(OPT(\mathcal{F}_1)) + v(OPT(\mathcal{F}_2))$, and we obtain a solution of sum at most $\alpha \cdot \varphi \cdot [v(OPT(\mathcal{F}_1)) + v(OPT(\mathcal{F}_2))]$. The worst case ratio of our solution is the maximum of $\alpha \cdot (1 + \frac{1}{\varphi})$ and $\alpha \cdot \varphi$, which are both equal to $\alpha \cdot \varphi$. □

Theorem 4.2 *If ALG α-approximates Π, we can 4α-approximate sum and ratio Π^k.*

[2]The golden ratio is the solution to $\varphi = \varphi^2 - 1$, or approximately 1.6.

Proof. Run *ALG* on each of the k single-level problems contained within Π^k. We cluster these solutions into intervals so that the last solution in each interval is at most twice the cost of the first solution in the same interval. The solution for level ℓ will be the last solution in ℓ's interval together with the last solutions of all prior intervals.

Formally, let A_ℓ denote ALG's solution for level ℓ and v_ℓ its cost. Observe that $v_\ell = v(A_\ell) \leq \alpha \cdot v(OPT(\mathcal{F}_\ell))$. Define interval i as all levels ℓ with $2^{i-1}v_1 \leq v_\ell < 2^i v_1$. Thus interval 1 contains ℓ with $v_1 \leq v_\ell < 2v_1$, interval 2 contains ℓ with $2v_1 \leq v_\ell < 2^2 v_1$, and so on. Let $max(i)$ denote the last level in interval i, as illustrated in Figure 2.

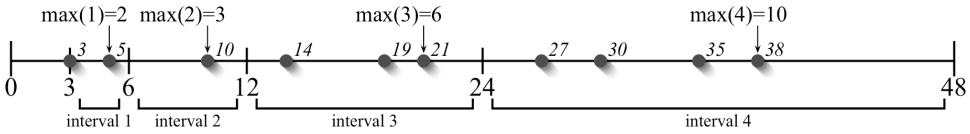


Figure 2: Ten-level incremental input clustered into four intervals. Black dots represent single-level solutions, labeled by cost and sorted on a number line. All $max(i)$ levels are indicated.

For each level ℓ in interval i, define $S_\ell = \bigcup_{j=1}^{i} A_{max(j)}$. Repeat for all levels to yield $\mathbf{S} = (S_1, S_2, \ldots, S_k)$, which is incremental by construction. Note that $\mathbf{S}$ is also feasible because, by monotonicity, any superset of $A_{max(i)}$ is feasible for all levels in interval i. To establish the approximation bound, recall that $2^{i-1}v_1 \leq v_\ell < 2^i v_1$. Then

$$\begin{array}{rcccccl} v(S_\ell) & \leq & \sum_{j=1}^{i} v_{max(j)} & < & \sum_{j=0}^{i} 2^j v_1 & < & 2^{i+1}v_1 \\ & = & 4 \cdot 2^{i-1} v_1 & \leq & 4 \cdot v_\ell & \leq & 4 \cdot \alpha \cdot v(OPT(\mathcal{F}_\ell)). \end{array}$$ □

It is worth comparing the performance of these two schemes to that of the general $(\frac{\alpha}{\mathcal{H}_k})$-approximation algorithm[3] for max sum incremental packing problems given in [8]. First, the above results apply to both min sum and min ratio problems, whereas there are no known approximation schemes for the max ratio metric. Next, [8] presents a $(\frac{\alpha}{1.5})$-approximation for 2-level max sum incremental packing problems, which is slightly better than the $(\varphi \cdot \alpha)$-approximation provided by Theorem 4.1. For large k, however, we have a (4α)-approximation for any incremental covering problem, much better than the $O(\frac{\alpha}{\log k})$-approximation for max sum incremental packing problems. This approximation gap can only get wider, as the latter algorithm is tight for incremental flow, whereas there is no evidence that the former algorithm cannot be improved upon. If we fix k, both max sum packing and min sum covering problems can be approximated within a constant factor of the best single-level approximation.

Concerning the problems introduced in Section 3, Theorems 4.1-4.2 yield respectable constant factor approximation algorithms for the NP-hard incremental problems IC and min ratio IEC. The φ-approximation to two-level min ratio incremental cut contrasts sharply with what is known about the max ratio incremental version of min cut's dual, incremental flow, which cannot be approximated better than $\omega(\frac{1}{n})$ [7]. Thus the incremental versions of dual problems are not themselves duals. Lastly, Theorem 4.2 and the $\log n$-approximation for r-domination [1] yield a $4 \log n$-approximation for incremental r-domination – a new problem and solution which arise naturally from our general model for minimization problems.

[3] $\mathcal{H}_k = \frac{1}{1} + \frac{1}{2} + \ldots + \frac{1}{k}$ is the kth harmonic number, which grows like $\Theta(\log k)$.

5 Online Algorithms Revisited

As observed in the introduction, incremental problems provide insight into the complexity of online problems. To illustrate and refine this concept, we demonstrate how the study of incremental Steiner tree leads to a deeper comprehension of its online version's results.

Given an undirected graph $G = (V, E)$ with edge weights and a terminal set $T \subseteq V$, the *Steiner tree* problem finds a minimum cost tree that spans all of T. The best-known polynomial-time algorithm for Steiner tree is currently a 1.55-approximation [24]. In *online Steiner tree*, terminal nodes arrive one at a time and the solution tree is expanded in discrete steps to include each new terminal [25, 26]. Alon and Azar [25] show an almost tight lower bound of $\Omega(\frac{\log n}{\log \log n})$ for the competitive ratio of any online algorithm for Steiner Tree. *Incremental Steiner tree* (IST) is quite similar to the online case. Given G and a sequence of terminal sets $T_1, T_2, \ldots, T_k$, IST finds a sequence of trees $S_1 \subseteq S_2 \subseteq \cdots \subseteq S_k$ such that S_ℓ spans T_ℓ.

Steiner tree belongs to a class of covering problems that do not quite fit our general model. Solutions to Steiner tree must be minimal covers of a target set, i.e. trees. This violates monotonicity, as supersets of trees might not be trees and hence might not be feasible. One way to circumvent this issue is to disregard the minimality condition and continue on with our general model. It is perhaps more desirable, however, to relax monotonicity. For the purpose of the proofs and algorithms given in this paper, it is sufficient to require only the following. If S is a minimal cover of T and S' is a mimimal cover of $T' \subseteq T$, then there exists a minimal U covering T such that $S' \subseteq U \subseteq S' \cup S$. I.e. we can augment any subsolution into a feasible solution using only objects of another given feasible solution. Observe that $v(U) \leq v(S') + v(S)$. Steiner tree exhibits this modified monotonicity property, therefore Theorem 4.2 gives a $(4 \cdot 1.55)$-approximation to k-level max ratio IST.

Considering the similarity between online Steiner tree and IST, this result is actually quite revealing. Recall from Section 1 the two factors that contribute to the hardness of an online problem: incremental constraints and adversarial constraints. The approximation gap between IST and online Steiner tree reveals that the hardness of the latter has little to do with the incremental constraint. Whereas online Steiner tree cannot be approximated better than $\Omega(\frac{\log n}{\log \log n})$, IST has a constant-factor approximation. Thus adding the adversarial constraint increases the problem's complexity, and we can attribute the hardness of approximation of online Steiner tree to its adversarial constraint. In this way, our general model has the potential to offer new insights into the complexity of previously studied online problems.

6 Extensions

The large field of related work discussed in Section 1 motivates many interesting extensions to the results discussed in this paper. Our incremental model could be extended to handle incomplete knowledge of future constraints, such as with online and stochastic problems. It is worth investigating a model that relaxes the incremental constraint but charges some price for every violation, as seen in online bipartite matching [14]. Alternatively, one could relax the covering constraint but charge some price for each element left uncovered, as in facility location with outliers [27]. Lastly, any given optimization problem has many potential incremental variants, only a few of which were discussed in this paper.

References

[1] J. Bar-Ilan, G. Kortsarz, and D. Peleg, "How to allocate network centers," *J. Algorithms*, vol. 15, no. 3, pp. 385–415, 1993.

[2] R. R. Mettu and C. G. Plaxton, "The online median problem," in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 2000, p. 339.

[3] C. G. Plaxton, "Approximation algorithms for hierarchical location problems," in *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*. ACM Press, 2003, pp. 40–49.

[4] T. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoretical Computer Science*, vol. 38, pp. 293–306, 1985.

[5] G. Lin, C. Nagarajan, R. Rajaraman, and D. P. Williamson, "A general approach for incremental approximation and hierarchical clustering," in *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2006.

[6] S. Dasgupta, "Performance guarantees for hierarchical clustering," in *Proceedings of the 15th Annual Conference on Computational Learning Theory*. Springer-Verlag, 2002, pp. 351–363.

[7] J. Hartline and A. Sharp, "Hierarchical flow," in *Proceedings of the 2nd International Network Optimization Conference*, 2005, pp. 681–687.

[8] ——, "An incremental model for combinatorial maximization problems," in *Proceedings of the 5th International Workshop on Experimental Algorithms*. Springer-Verlag, 2006, pp. 36–48.

[9] A. Borodin and R. El-Yaniv, *Online computation and competitive analysis*. New York, NY, USA: Cambridge University Press, 1998.

[10] A. Fiat and G. J. Woeginger, Eds., *Online Algorithms, The State of the Art*, ser. Lecture Notes in Computer Science, vol. 1442. Springer, 1998.

[11] D. D. Sleator and R. E. Tarjan, "Amortized efficiency of list update and paging rules," *Commun. ACM*, vol. 28, no. 2, pp. 202–208, 1985.

[12] E. G. Coffman, M. R. Garey, and D. S. Johnson, "Dynamic bin packing," *SIAM Journal on Computing*, vol. 12, no. 2, pp. 227–258, 1983.

[13] A. Gyárfás and J. Lehel, "Online and first-fit colorings of graphs," *J. Graph Th.*, vol. 12, pp. 217–227, 1988.

[14] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*. ACM Press, 1990, pp. 352–358.

[15] H. Kaplan and M. Szegedy, "On-line complexity of monotone set systems," in *Proceedings of the 10th annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1999, pp. 507–516.

[16] A. Gupta, M. Pál, R. Ravi, and A. Sinha, "Boosted sampling: approximation algorithms for stochastic optimization," in *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*. ACM Press, 2004, pp. 417–426.

[17] N. Immorlica, D. Karger, M. Minkoff, and V. S. Mirrokni, "On the costs and benefits of procrastination: approximation algorithms for stochastic combinatorial optimization problems," in *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2004, pp. 691–700.

[18] B. C. Dean, M. X. Goemans, and J. Vondrak, "Adaptivity and approximation for stochastic packing problems," in *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2005, pp. 395–404.

[19] S. A. Plotkin, D. B. Shmoys, and É. Tardos, "Fast approximation algorithms for fractional packing and covering problems," in *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 1991, pp. 495–504.

[20] N. Garg and J. Koenemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems." in *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 1998, p. 300.

[21] J. Kleinberg and E. Tardos, *Algorithm Design*. Addison-Wesley, 2005.

[22] O. Parekh, "Edge dominating and hypomatchable sets," in *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2002, pp. 287–291.

[23] C. Gavoile, D. Peleg, A. Raspaud, and E. Sopena, "Small k-dominating sets in planar graphs with applications," in *Proc. Workshop on Graphs, WG'01, LNCS 2204*, 2001, pp. 201–216.

[24] G. Robins and A. Zelikovsky, "Improved steiner tree approximation in graphs," in *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2000, pp. 770–779.

[25] N. Alon and Y. Azar, "On-line steiner trees in the euclidean plane," in *Proceedings of the 8th Annual Symposium on Computational Geometry*. ACM Press, 1992, pp. 337–343.

[26] A. Meyerson, "Online algorithms for network design," in *Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures*. ACM Press, 2004, pp. 275–280.

[27] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers," in *Symposium on Discrete Algorithms*, 2001, pp. 642–651.

[28] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis, "The complexity of multiterminal cuts," *SIAM J. Comput.*, vol. 23, no. 4, pp. 864–894, 1994.

[29] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, 1972, pp. 85–103.

A Incremental Cut

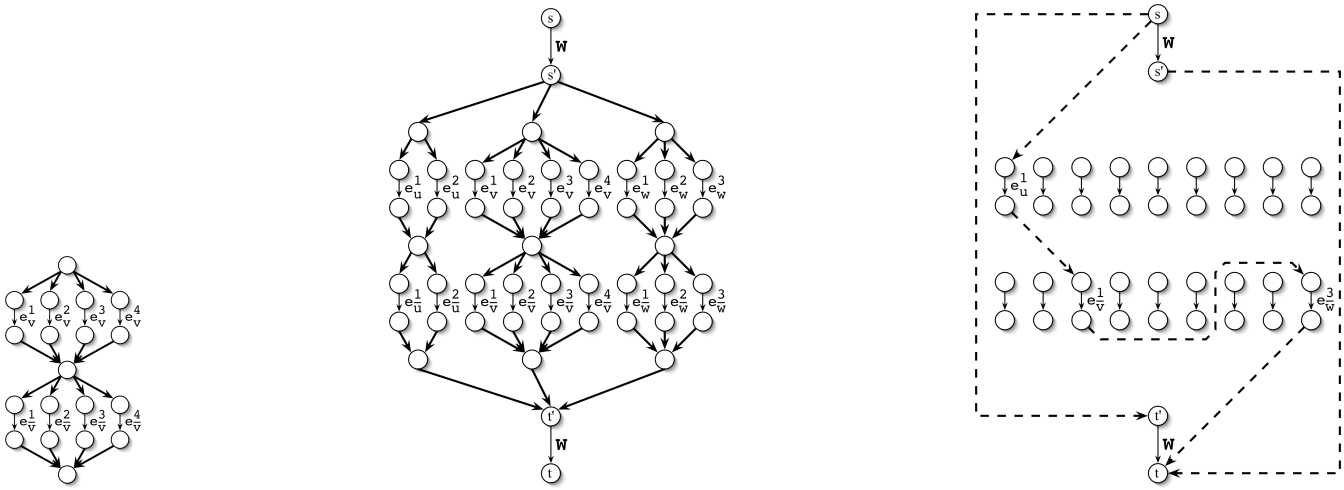
Theorem A.1 *Min ratio, min sum, and demand IC are NP-hard for directed unweighted graphs, $k \geq 2$.*

Theorem A.1 follows via a reduction from 3-SAT. Given a formula ϕ with n variables and m clauses, we construct an incremental cut instance for which solutions of a certain ratio and sum are feasible if and only if ϕ is satisfiable. The demand version follows as a special case.

3-SAT Reduction. Create source s, sink t, auxiliary vertices s', t' and edges (s, s'), (t', t) of weight $W > 0$. Let $|v|$ denote the number of appearances of variable v or its negation $\overline{v}$. For each v, create $2|v|$ unit-weight edges $e_v^1, e_v^2, \ldots, e_v^{|v|}, e_{\overline{v}}^1, e_{\overline{v}}^2, \ldots, e_{\overline{v}}^{|v|}$ linked by high-weight edges as shown in Figure 3(a). Connect all variable gadgets in parallel between s' and t' using high-weight edges to form the level one graph G_1 in Figure 3(b).

Next we construct the level two graph G_2. First we add high-weight bypass edges (s, t') and (s', t). Then for each clause c, we create an s-t path that passes in series through some e_ℓ^i for each of the three literals ℓ in c. These paths are such that every e_ℓ^i edge is used by at most one clausal path, and non-e_ℓ^i edges are given high weight. The bypass edges and one example clausal path are illustrated in Figure 3(c). We give our high-weight edges cost $20m$, thereby preventing their use in any reasonable-cost solution. Finally, replace weight w edges with w unit-weight parallel paths.

Set $W = 6m$. Then all minimum cuts of G_1 have cost $3m$ and contain either all e_v^i edges or all $e_{\overline{v}}^i$ edges for each variable v; the only other reasonable-cost minimal cuts are $\{(s, s')\}$ and $\{(t', t)\}$ which cost $W > 3m$. Moreover, all reasonable-cost cuts of G_2 contain both (s, s') and (t', t). G_2 can be cut



(a) (b) (c)

Figure 3: 3(a): A variable gadget. 3(b): The level one graph. 3(c): The level two graph, showing bypass edges and a clausal path for clause $c = \{u, \overline{v}, \overline{w}\}$. Bold edges have prohibitively high cost. Thin edges have unit cost unless labelled otherwise. Solid edges are level one edges whereas dashed edges are level two edges.

optimally at cost $2W + m = 13m$ by cutting (s, s'), (t', t), and the first unit cost edge in each of the m clausal paths. Lemma A.2 completes the proof.

Lemma A.2 *ϕ is satisfiable iff (G_1, G_2) has an incremental cut of ratio $\frac{15}{13}$ or sum $18m$.*

[$\Rightarrow$] Given a satisfying assignment A, we construct an incremental cut as follows: if $A(v) = true$ then cut all e_v^i, otherwise cut all $e_{\overline{v}}^i$. This selection of $3m$ edges is a cut of G_1. To cut G_2 we only add (s, s') and (t', t). This costs an additional $2W = 12m$, yielding an incremental cut with ratio $\max(\frac{3m}{3m}, \frac{15m}{13m}) = \frac{15}{13}$ and sum $3m + 15m = 18m$. We claim these edges are sufficient to cut s from t in G_2: if not then some s-t path would remain. Because (s, s') is cut, this path must originate along one of the m clausal paths. It cannot follow such a path all the way from s to t, as each clause contains one true literal whose edge is contained in our G_1 cut. On the other hand, any deviation from a clausal path is only possible immediately after the path passes through an e_v^i or $e_{\overline{v}}^i$ edge. If it deviates after an edge of the form $e_{\overline{v}}^i$ then the only path remaining to t passes through the cut edge (t', t), a contradiction. And if it deviates after an edge of the form e_v^i then all $e_{\overline{v}}^i$ must be cut, and it is impossible to exit the variable gadget except to follow the clausal path.

[$\Leftarrow$] Now suppose we are given an incremental cut (S_1, S_2) of ratio $\frac{15}{13}$ or sum $18m$. Without loss of generality we may assume that the cut S_1 is minimal. Furthermore, we claim that neither (s, s') nor (t', t) is contained in S_1; if they were then our level one ratio would be at least $\frac{6m}{3m} = 2 > \frac{15}{13}$ and our total sum at least $6m + 13m = 19m > 18m$. Thus the cut S_1 must contain either all e_v^i or $e_{\overline{v}}^i$ edges for each variable v, at cost $3m$. This defines our truth assignment A: set $A(v) = true$ if all e_v^i are cut and $A(v) = false$ if all $e_{\overline{v}}^i$ are cut. In addition to S_1, the cut S_2 contains only (s, s') and (t', t); if it contained even one more edge, it would have ratio $\frac{12m+3m+1}{12m+m} > \frac{15}{13}$ and sum $3m + 15m + 1 > 18m$. Hence the addition of (s, s') and (t', t) to S_2 must suffice to cut s from t in G_2, and all clausal paths are cut by our level one cut, indicating that under our assignment every clause contains at least one true literal. $\square$

Theorem A.3 *Demand IC is NP-hard for undirected unweighted graphs, $k \geq 2$.*

The proof of Theorem A.3 follows via a reduction from multiway cut (MWC) with unit weights and 3 terminals, which is known to be NP-hard [28]. Given an undirected graph $G = (V, E)$ and terminal set $T = \{t_1, t_2, t_3\}$, a *multiway cut* is a set of edges whose removal disconnects the terminals from each other. Given an integer $C \geq 1$, MWC asks whether there exists such a set of size at most C. Given an MWC instance $(G, \{t_1, t_2, t_3\}, C)$, we construct an instance (G_1, G_2) of undirected IC with demands that is feasible if and only if G has a multicut of capacity at most C.

To this end, define G_1 as G augmented with super-source s, super-sink t, edges $\{t_2, t\}$ and $\{t_3, t\}$ of weight $C + 1$, and edge $\{s, t_1\}$ of weight $3C$. G_2 augments G_1 with edge $\{s, t_2\}$ of weight $3C$. The demands are $d_1 = C$ and $d_2 = 2C + 1$. With these demands, the $3C$ edges effectively have infinite weight and cannot be part of any feasible solution. Further observe that any feasible level 1 cut cannot cut either of the $(C + 1)$-cost edges, whereas all feasible level 2 cuts must cut $\{t_2, t\}$ but cannot cut any other weighted edge. We convert G_1 and G_2 into unweighted graphs by replacing weight $w > 1$ edges with w unit-weight parallel paths. Lemma A.4 completes the proof.

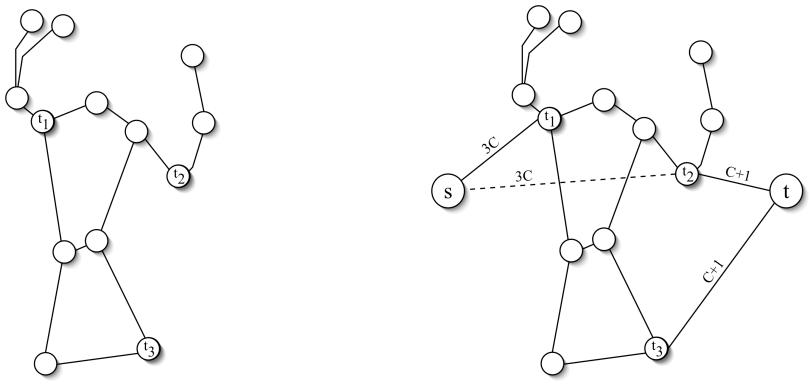


Figure 4: An unweighted MWC instance with $C = 3$ and its corresponding 2-level undirected IC instance.

Lemma A.4 *There exists a multiway cut of size at most C in G if and only if there exists an incremental cut of (G_1, G_2) satisfying demands (d_1, d_2).*

[$\Rightarrow$] Suppose there is a multiway cut $S \subseteq E$ such that $v(S) \leq C$. Define $\mathbf{S} = (S_1, S_2) = (S, S \cup \{\{t_2, t\}\})$. Certainly $\mathbf{S}$ is a feasible solution: $S_1 \subseteq S_2$, $S_\ell \subseteq E_\ell$, and no s-t paths exist in $G_1 - S_1$ and $G_2 - S_2$. Lastly, the cut costs satisfy our demands.

[$\Leftarrow$] Now suppose there is an incremental solution (S_1, S_2) such that $v(S_1) \leq C$ and $v(S_2) \leq 2C + 1$. Define edge set $S = S_2 \cap E$, i.e. the edges of G appearing in the level 2 cut. Recall that $\{t_2, t\}$ of weight $C + 1$ must be an element in S_2, and therefore S, which has $\{t_2, t\}$ removed, has cost $v(S) \leq C$. Furthermore, we claim that S is a multiway cut. The set S_1 must cut all t_1-t_2 and t_1-t_3 paths to separate s from t in G_1 without cutting $\{s, t_1\}$, $\{t_2, t\}$, or $\{t_3, t\}$. For analogous reasons, S_2 must cut all t_2-t_3 paths, thereby completing the multiway cut. □

B Incremental Edge Cover

Theorem B.1 *Min ratio IEC is NP-hard for two levels.*

We prove Theorem B.1 by reduction from partition, an NP-hard problem [29]. Given a finite set A and sizes $s(a) \in \mathbb{Z}^+$ for all $a \in A$, a *partition* of A is some $A' \subseteq A$ such that $\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)$. We construct a 2-level instance of IEC for which a ratio of $(1+\varphi)/2$ is achievable if and only if A has a partition, where φ is the golden ratio.

Begin with a single vertex s and, for each $a \in A$, add vertices u_a, v_a and edges $e_a^1 = \{s, u_a\}$ and $e_a^2 = \{u_a, v_a\}$ with respective costs $s(a)$ and $\varphi \cdot s(a)$. This is our IEC graph G. Define target sets $T_1 = \bigcup_{a \in A} u_a$ and $T_2 = \bigcup_{a \in A} \{u_a, v_a\}$, as shown in Figure 5. Let $S = \sum_{a \in A} s(a)$. The optimal level one cover S_1^* is all e_a^1 edges and has cost S. The optimal level two cover S_2^* is all e_a^2 edges and has cost $\varphi \cdot S$.

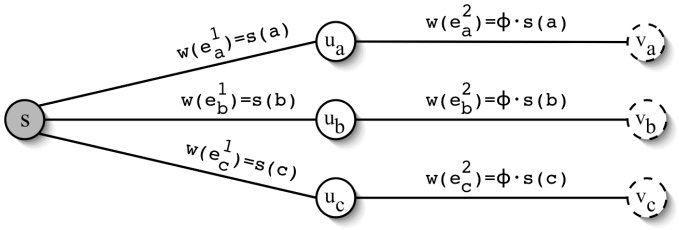


Figure 5: The two-level IEC graph for an instance of partition with $A = \{a, b, c\}$. Solid nodes must be covered at level one. Dashed nodes must be covered at level two. Vertex s need not be covered.

Theorem B.1 follows from Lemma B.2. Min ratio IEC on weight-bounded graphs is still an open problem.

Lemma B.2 *There is a partition of A if and only if there is a $\frac{1+\varphi}{2}$-ratio cover of G.*

[$\Rightarrow$] Given a partition A' of A, construct cover (S_1, S_2) by selecting $S_1 = \{e_a^1 \mid a \in A'\} \cup \{e_a^2 \mid a \in A \setminus A'\}$ and $S_2 = S_1 \cup \{e_a^2 \mid a \in A'\}$. This is a feasible solution, as both v_a and w_a are covered for each element $a \in A$. Furthermore,

$$\begin{aligned} r_1 = \tfrac{v(S_1)}{v(S_1^*)} &= \tfrac{\sum_{a \in A'} s(a) + \sum_{a \in A \setminus A'} \varphi s(a)}{S} \\ &= \tfrac{S/2 + \varphi \cdot S/2}{S} \\ &= \tfrac{1+\varphi}{2} \end{aligned} \qquad \begin{aligned} r_2 = \tfrac{v(S_2)}{v(S_2^*)} &= \tfrac{\varphi \cdot S + \sum_{a \in A'} s(a)}{\varphi \cdot S} \\ &= \tfrac{\varphi \cdot S + S/2}{\varphi \cdot S} \\ &= \tfrac{1+\varphi}{2}. \end{aligned}$$

[$\Leftarrow$] Given a $\frac{1+\varphi}{2}$-ratio cover (S_1, S_2), we claim that $v(S_2) = \varphi \cdot S + S/2$. If $v(S_2) > \varphi \cdot S + S/2$ then $r_2 > \frac{1+\varphi}{2}$, a contradiction. Otherwise, suppose $v(S_2) < \varphi \cdot S + S/2$. Cover S_2 must contain all e_a^2 edges, and therefore the cost of all e_a^1 edges in S_2 is $S/2 - \epsilon$ for some $\epsilon > 0$. However, for each e_a^1 not in S_2, e_a^2 must be in S_1, so the cost of all e_a^2 edges in S_1 is at least $\varphi \cdot (S/2 + \epsilon)$. This yields $v(S_1) > (S/2 - \epsilon) + \varphi \cdot (S/2 + \epsilon) > S/2 + \varphi \cdot S/2$ and hence $r_1 > \frac{1+\varphi}{2}$, a contradiction. Thus $A' = \{a \mid e_a^1 \in S_2\}$ is a partition of A. □

Lemma B.3 *For every incremental cover there is a reduced cover of no higher cost.*

Proof. We show how to convert any incremental cover into a reduced cover of equal or lower cost. To this end, let $e = \{u, v\}$ be any edge in the cover. There are three cases:

(i) Neither u nor v are in T_k. Remove e to produce a feasible solution of lower cost.

(ii) Exactly one of u and v is in T_k. Without loss of generality, assume $u \in T_k$. If e is added to our cover after level ℓ_u, remove it, because u must be covered by some other edge at or prior to level ℓ_u. Otherwise, replace e with e_u at level ℓ_u.

(iii) Both u and v are in T_k. Without loss of generality, assume $\ell_u \leq \ell_v$. If e is added to our cover after level ℓ_v, remove it. If e appears at or before level ℓ_u, keep e in our cover, but remove it from all levels prior to ℓ_u. Otherwise, e appears after level ℓ_u but no later than level ℓ_v. In this case, replace e with e_v at level ℓ_v.

Executing these cases on all edges in the original cover neither affects the feasibility of the cover nor increases its cost. All edges in the produced cover are of the appropriate form and therefore the new cover is reduced. □

Lemma B.4 *There is a subset edge cover S of (G', T_k) of cost $v(S)$ if and only if there is a reduced incremental edge cover $\mathbf{S}$ of $(G, \mathbf{T})$ of cost $\mathcal{V}(\mathbf{S}) = v(S)$.*

[⇐] Given a reduced cover $\mathbf{S}$, we build subset edge cover S by considering each edge $e \in \mathbf{S}$. By definition, there are only two types of edges in a reduced cover. If $e = \{u, v\}$ for $u, v \in T_k$ appears in $\mathbf{S}$ at level $min(\ell_u, \ell_v)$, then include $\{u, v\}$ in S. Otherwise, $e = e_u$ for some $u \in T_k$ and appears at level ℓ_u. In this case, include $\{u, \hat{u}\}$ in S. Because $\mathbf{S}$ incrementally covers T_k, the cover S is indeed a cover of T_k. Furthermore, the cost of the two covers is edge-by-edge equivalent and therefore $v(S)$ is exactly $\mathcal{V}(\mathbf{S})$.

[⇒] For analogous reasons, any $\mathbf{S}$ produced as described in Algorithm IEC from some S that covers T_k will be a feasible reduced solution with cost $\mathcal{V}(\mathbf{S}) = v(S)$. □

Theorem B.5 *Min sum IEC is in P.*

Proof. The correctness of Algorithm IEC follows from Lemmas B.3-B.4. □