An Incremental Model for Combinatorial Minimization

Jeff Hartline Alexa Sharp

Cornell University, Ithaca, NY 14853
{jhartlin, asharpa}@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 give a general model for producing incremental versions of traditional covering problems and formalize several natural incremental metrics. We also demonstrate how to convert conventional algorithms into incremental algorithms with only a constant factor loss in approximation power. Lastly, we introduce incremental versions of min cut and edge cover, and discuss the application of the incremental model to $(k, r)$-center and Steiner tree.

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 $\ell$ years is not much more than the optimal number of towers needed to satisfy just the year $\ell$ 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. There is no parameter $k$, the client set grows at discrete time intervals, and 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 incrementally; 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 \( \alpha \)-competitive if this maximum is no more than \( \alpha \). Alternatively, if we are more interested in overall performance, our goal is to minimize the sum of the solutions over all time steps. If each tower has an annual operating cost, minimizing net cost over all years corresponds to minimizing this sum. Lastly, if we have specific demands, our goal is to obtain an incremental solution satisfying the required demand at each level.

Most combinatorial minimization problems can be extended into 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 but does not remove edges from any prior cut. In incremental edge cover, we are given an edge-weighted graph and a increasing sequence of target vertex sets. We want an edge cover for each target set, such that each cover builds on all prior covers.

**Related Work.** Recent interest in incremental optimization has focused predominantly on incremental versions of classic NP-hard problems. 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 \( \alpha \)-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.

In the area of polynomial-time problems, Hartline and Sharp [7] introduce incremental versions of max flow, and find the first instance of a polynomial-time problem whose incremental version is NP-hard. In [8] we present a general model and algorithmic framework for combinatorial packing problems, achieving an \( O(\alpha/\log k) \)-approximation for any \( k \)-level incremental packing problem with an \( \alpha \)-approximation algorithm. This algorithm is tight in the case of maximum flow, however we show that better algorithms exist for incremental bipartite matching and incremental knapsack.

Online problems share many similarities with the incremental model. For instance, their input arrives over time and their solutions build on each other incrementally. However, online algorithms act with no knowledge of future input and are evaluated only on their final output [9, 10]. One compares the final solution built up over time to the best offline solution for this last level. Since the online solution is constrained to keep all intermediate solutions, it is more reasonable to compare the complete online solution sequence with the best incremental solution. In particular, it may not be possible to obtain reasonable solutions at each level while simultaneously guaranteeing a final solution with good competitive ratio. Therefore the competitive ratio may be an overly harsh judgment of an online algorithm attempting to be fair at each time step. Online algorithms have been studied in many contexts, including bin packing[11], graph coloring [12], and bipartite matching [13]. Analysis of these online algorithms could benefit from theoretical results on the corresponding offline incremental problem. We will discuss some of these issues in Section 5.2.

Stochastic optimization also resembles our incremental framework, in that instances have multi-stage input and incremental solutions. However, the input is not fully known
at the outset, and the goal is to find a final solution of minimum cost. We motivate our
general models by those developed for stochastic problems [14, 15, 16]. General models
for single-level optimization problems are available in [17, 18].

Our Results. We introduce a general incremental model 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. Our central
contribution is a general technique to translate exact or approximate algorithms for non-
incremental minimization problems into approximation algorithms for the corresponding
incremental versions. Lastly, we discuss the implications of applying our model to (k, r)-
center and Steiner tree.

The incremental model is laid out in Section 2, along with two general approximation
techniques. We present incremental cut in Section 3 and incremental edge cover in
Section 4. In Section 5, we discuss (k, r)-center and Steiner tree.

2 General Incremental Model for Covering Problems

2.1 A Framework for Optimization Problems

Single-Level Problems. We represent a single-level abstract optimization problem
π as a tuple (X, F, v). The set X is a collection of objects, and feasible solutions are
subsets of X that collectively achieve a certain goal [19]. The set of all such solutions
is $F \subseteq 2^X$, and $v : F \rightarrow \mathbb{R}$ 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 $v(S) = \sum_{x \in S} w_x$. The optimal solution $OPT(X, F, v)$,
or $OPT(F)$ if $X$ and $v$ are understood, is a subset $S \subseteq X$, $S \in F$ optimizing $v(S)$.

This notation is from [8], which adapts the general models of [14, 15]. This paper
focuses on covering problems, a class of minimization problems that exhibit monotonic-
ity: $S \in F$ and $S' \supseteq S$ imply $S' \in F$. In other words, any superset of a feasible solution
is also feasible. Min cut, edge cover, vertex cover, and many other classic minimization
problems all fall into this framework. See [8] for results on packing problems.

Incremental Problems. We extend this notion of an optimization problem to a
multi-level incremental optimization problem. The $k$-level incremental version of $\pi$
denoted $\pi^k$, consists of $k$ instances of $\pi$ with the same object set $X$ and cost function
$v$, but each with its own feasible set. Thus an instance of $\pi^k$ is represented as a
tuple $(X, (F_1, F_2, \ldots, F_k), v)$. A solution is a tuple $S = (S_1, S_2, \ldots, S_k)$ that meets the
feasibility constraint $S_\ell \in F_\ell$ and the incremental constraint $S_\ell \subseteq S_{\ell+1}$. We further
assume $F_1 \supseteq F_2 \supseteq \ldots \supseteq F_k$, a reasonable assumption for covering problems.

There are several objective functions for incremental problems. The minimum ratio
objective is to obtain the smallest possible ratio of each level’s optimal solution: find
$S$ minimizing $R(S) = \max_{\ell} \frac{v(S_\ell)}{v(OPT(X, F_\ell, v))}$. This is the same as the competitive ratio
of 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 $S$ minimizing $V(S) = \sum_\ell v(S_\ell)$. The demand
objective takes demands $d_1, \ldots, d_k$ and seeks an incremental solution with $v(S_\ell) \leq d_\ell$. 
We now consider two well-known problems, and demonstrate how they fit into this framework. There are multiple ways to define incremental versions of these problems, but we introduce only those subject to discussion in this paper.

2.1 Minimum 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 edge sets that cut $s$ from $t$. The cost of a cut $S$ is the sum of the weights of the edges it contains: $v(S) = \sum_{e \in S} w_e$. Incremental cut 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 $S_\ell$ is an $s$-$t$ cut of $(G, E_\ell)$ and $S_\ell \subseteq S_{\ell+1}$. $G$ may be directed or undirected in either of the above problems.

2.1.1 Minimum Cut

Along with $T$.

2.1.2 Subset 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 $v(S) = \sum_{e \in S} w_e$. 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 in [20]. 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 [20]. We define incremental edge cover 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 $S_\ell$ is a subset edge cover of $T_\ell$ in $G$, and $S_\ell \subseteq S_{\ell+1}$.

2.2 Approximation Algorithms for Incremental Minimization

Based on the model introduced in Section 2.1, we give general methods to convert single-level algorithms into incremental algorithms for the min sum and min ratio metrics.

**Theorem 2.1** If algorithm ALG $\alpha$-approximates $\Pi$, we can $(\phi \cdot \alpha)$-approximate min sum and min ratio versions of $\Pi^2$, where $\phi$ is the golden ratio$^1$.

**Proof.** Run ALG on both single-level input to obtain $v(ALG(F_1)) \leq \alpha \cdot v(OPT(F_1))$ and $v(ALG(F_2)) \leq \alpha \cdot v(OPT(F_2))$. We define incremental solution $(S_1, S_2)$ as follows:

(i) If $v(ALG(F_2)) \geq \phi \cdot v(ALG(F_1))$, then $S_1 = ALG(F_1)$, $S_2 = S_1 \cup ALG(F_2)$, so

\[
\begin{align*}
v(S_1) &= v(ALG(F_1)) \\
&\leq \alpha \cdot v(OPT(F_1)) \\
v(S_2) &\leq v(ALG(F_1)) + v(ALG(F_2)) \\
&\leq \frac{1}{\phi} \cdot v(ALG(F_2)) + v(ALG(F_2)) \\
&\leq (1 + \frac{1}{\phi}) \cdot \alpha \cdot v(OPT(F_2)).
\end{align*}
\]

(ii) If $v(ALG(F_2)) \leq \phi \cdot v(ALG(F_1))$, then $S_1 = ALG(F_2) = S_2$, so

\[
\begin{align*}
v(S_1) &= v(ALG(F_2)) \\
&\leq \phi \cdot v(ALG(F_1)) \\
v(S_2) &= v(ALG(F_2)) \\
&\leq \phi \cdot \alpha \cdot v(OPT(F_1)).
\end{align*}
\]

$^1$The golden ratio is the solution to $\phi = \phi^2 - 1$, or approximately 1.6.
The optimal sum is at most \(v(OPT(F_1)) + v(OPT(F_2))\), and we obtain a solution of sum at most \(\alpha \cdot \phi \cdot (v(OPT(F_1)) + v(OPT(F_2)))\). The worse case ratio of our solution is the maximum of \(\alpha \cdot (1 + \frac{1}{\phi})\) and \(\alpha \cdot \phi\), which are both equal to \(\alpha \cdot \phi\).

**Theorem 2.2** If ALG \(\alpha\)-approximates II, we can \(4\alpha\)-approximate sum and ratio \(\Pi^k\).

**Proof.** Run ALG on each of the \(k\) single-level problems contained within \(\Pi^k\). 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 \(\ell\) will be the last solution in \(\ell\)'s interval together with the last solutions of all prior intervals.

Formally, let \(A_{\ell}\) denote ALG’s solution for level \(\ell\) and \(v_{\ell}\) its cost. Observe that \(v_{\ell} = v(A_{\ell}) \leq \alpha \cdot v(OPT(F_\ell))\). Define interval \(i\) as all levels \(\ell\) with \(2^{i-1}v_1 \leq v_{\ell} < 2^iv_1\). Thus interval 1 contains \(\ell\) with \(v_1 \leq v_{\ell} < 2v_1\), interval 2 contains \(\ell\) with \(2v_1 \leq v_{\ell} < 2^2v_1\), and so on. Let \(\max(i)\) denote the last level in interval \(i\), as illustrated in Figure 1.

![Figure 1: 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.](image)

For each level \(\ell\) in interval \(i\), define \(S_\ell = \bigcup_{j=1}^i A_{\max(j)}\). Repeat for all levels to yield \(S = (S_1, S_2, \ldots, S_k)\), which is incremental by construction. Note that \(S\) 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 \(2^{i-1}v_1 \leq v_{\ell} < 2^iv_1\). Then
\[
v(S_\ell) \leq \sum_{j=1}^i v_{\max(j)} < \sum_{j=0}^{2^i-1} 2^jv_1 < 2^{i+1}v_1
\]
\[
\begin{align*}
&= 4 \cdot 2^i - 1v_1 \\
&\leq 4 \cdot v_{\ell} \\
&\leq 4 \cdot \alpha \cdot v(OPT(F_\ell)).
\end{align*}
\]

**3 Incremental Cut**

It is well-known that directed and undirected min cut have polynomial-time algorithms. In contrast, directed and undirected incremental cut (IC) are intractable.

**3.1 Directed Incremental Cut**

**Theorem 3.1** Min ratio and min sum IC are NP-hard for unweighted directed graphs.

Theorem 3.1 follows via a reduction from 3-SAT. Given a formula \(\phi\) 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 \(\phi\) is satisfiable.

**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 \(e_v^1, e_v^2, \ldots, e_v^{2|v|}, e_{\overline{v}}^1, e_{\overline{v}}^2, \ldots, e_{\overline{v}}^{2|v|}\) linked by high-weight edges as shown in Figure 2(a). Connect all variable gadgets in
Figure 2: 2(a): A variable gadget. 2(b): The level one graph. 2(c): The level two graph, showing bypass edges and a clausal path for clause \( c = \{u, v, w\} \). 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.

parallel between \( s' \) and \( t' \) using high-weight edges to form the level one graph \( G_1 \) in Figure 2(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 \( e^i_\ell \) for each of the three literals \( \ell \) in \( c \). These paths are such that every \( e^i_\ell \) edge is used by at most one clausal path, and non-\( e^i_\ell \) edges are given high weight. The bypass edges and one example clausal path are illustrated in Figure 2(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 \( e^i_v \) edges or all \( e^j_v \) 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 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 3.2** \( \phi \) is satisfiable iff \((G_1, G_2)\) has an incremental cut of ratio \( \frac{15}{13} \) or sum \( 18m \).

\[ \implies \] Given a satisfying assignment \( A \), we construct an incremental cut as follows: if \( A(v) = true \) then cut all \( e^i_v \), otherwise cut all \( e^j_v \). 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 $e_i^t$ or $e_i^b$ edge. If it deviates after an edge of the form $e_i^t$, 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 $e_i^b$ then all $e_i^b$ must be cut, and it is impossible to exit the variable gadget except to follow the clausal path.

[⇒] 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 $\frac{6m}{2m} = 2 > \frac{15}{13}$ and our total sum at least $6m + 13m = 19m > 18m$. Thus the cut $S_1$ must contain either all $e_i^t$ or $e_i^b$ edges for each variable $v$, at cost $3m$. This defines our truth assignment $A$: set $A(v) = \text{true}$ if all $e_i^t$ are cut and $A(v) = \text{false}$ if all $e_i^b$ 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 $\frac{12m + 3m + 1}{12m + m + 1} > \frac{15}{13}$ 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.

Theorem 3.1 follows from Lemma 3.2. Because min ratio IC is a special case of IC with demands, Theorem 3.1 implies Corollary 3.3.

**Corollary 3.3** Directed unweighted IC with demands is NP-hard for $k \geq 2$.

### 3.2 Undirected Incremental Cut

**Theorem 3.4** Undirected unweighted IC with demands is NP-hard for $k \geq 2$.

**Proof.** See Appendix A.

### 4 Incremental Edge Cover

Section 3 establishes that, regardless of metric, IC is NP-hard. Analogous results for incremental max flow are presented in [7, 8]. This section shows that min sum incremental edge cover (IEC) is in P, whereas min ratio IEC is NP-hard.

#### 4.1 Minimum Sum Edge Cover

We begin with some notation. A vertex $t \in T_k$ is a level $\ell$ vertex if $t$ first appears in target set $T_\ell$; we denote the level of $t$ as $\ell_t$. For each 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 $\ell_t$. The algorithm follows.

**Algorithm IEC.** Given an IEC instance $(G, w, T_k)$, we construct a single-level subset edge cover instance $(G', w', T_k)$ such that covers of $G'$ are in one-to-one correspondence with reduced incremental covers of $G$. The construction occurs in two phases:

1. 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.\(^2\)

This construction is illustrated in Figure 3.

We solve this instance to obtain a min-cost subset edge cover \( S \), and build incremental solution \( S = (S_1, S_2, \ldots, S_k) \) 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 \( S_{\ell_u} \). Return \( S \) as an optimal incremental cover.

The correctness of Algorithm IEC follows from Lemmas 4.1-4.2. We refer the reader to Appendix B for the proof of Lemma 4.1, which shows that we can, without loss of generality, assume that the optimal incremental cover of \( G \) is a reduced cover. Lemma 4.2 confirms that optimal covers of \( G' \) correspond to optimal reduced covers of \( G \).

**Lemma 4.1** For every incremental cover there is a reduced cover of no higher cost.

**Lemma 4.2** 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 \( S \) of \((G, T)\) of cost \( V(S) = v(S) \).

\[ \Leftarrow \] Given a reduced cover \( S \), we build subset edge cover \( S \) by considering each edge \( e \in S \). 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 \( S \) 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 \( \ell_u \). In this case, include \( \{u, \hat{u}\} \) in \( S \). Because \( S \) 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 \( V(S) \).

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

Theorem 4.3 follows from the correctness and running time of Algorithm IEC.

**Theorem 4.3** Min sum IEC is in \( P \).

### 4.2 Minimum Ratio Edge Cover

**Theorem 4.4** Min ratio IEC is NP-hard for two levels.

\(^2\)Not all of these edges are necessary, but they do not increase the size of \( G' \) excessively.
We prove Theorem 4.4 by reduction from partition, an NP-hard problem [21]. 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 \((1 + \phi)/2\) is achievable if and only if \( A \) has a partition, where \( \phi \) is the golden ratio.

Begin with a single vertex \( s \) and, for each \( a \in A \), add vertices \( u_a, v_a \) and edges \( e_1^a = \{ s, u_a \} \) and \( e_2^a = \{ u_a, v_a \} \) with respective costs \( s(a) \) and \( \phi \cdot s(a) \). 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 4. Let \( S = \sum_{a \in A} s(a) \). The optimal level one cover \( S_1^* \) is all \( e_1^a \) edges and has cost \( S \). The optimal level two cover \( S_2^* \) is all \( e_2^a \) edges and has cost \( \phi \cdot S \).

**Figure 4:** 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.

**Lemma 4.5** There is a partition of \( A \) if and only if there is a \( \frac{1+\phi}{2} \)-ratio cover of \( G \).

Theorem 4.4 follows from Lemma 4.5, which we prove in Appendix B. Min ratio IEC on weight-bounded graphs is still an open problem.

## 5 Discussion

### 5.1 Incremental \((k, r)\)-Center

Recall the cell phone tower example 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:

(i) \textit{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) \textit{r-domination} [1, 22]: fix the radius \( r \) and cover all clients with as few centers as possible. The current best-known 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 centers such that, for any \( \ell \), the first \( \ell \) centers in the sequence are competitive with the optimal \( \ell \)-center solution. Gonzalez’ 2-approximation [4] is online, and thus applies to the incremental case. Unfortunately, \( k \)-center is not a covering problem and therefore our model cannot be applied.

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. Using the $\log n$-approximation of [1], Theorem 2.2 yields a $4\log n$-approximation to incremental $r$-domination. This new problem and its solution develop naturally from our general model for minimization problems.

5.2 Incremental Steiner Tree

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 [23]. 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 [24, 25]. Alon and Azar [24] show an almost tight lower bound of $\Omega(\log n/\log \log n)$ 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 $S_\ell$ spans $T_\ell$.

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 minimal 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 minimal subsolution into a feasible solution using only objects of a given feasible solution. Observe that $v(U) \leq v(S') + v(S)$. Steiner tree exhibits this modified monotonicity property, therefore Theorem 2.2 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. Two factors may contribute to the hardness of an online problem: incremental constraints and prediction constraints. In other words, do algorithms perform poorly because they must build on prior solutions or because they cannot predict future input? Both factors influence the hardness of an online problem, but the competitive ratio [26] fails to discern which is the more significant.

The approximation gap between IST and online Steiner tree reveals that the hardness of the latter has little to do with its incremental constraints. With the same constraints, IST has a constant-factor approximation. Online Steiner tree, however, cannot be approximated better than $\Omega(\log n/\log \log n)$, which we can now attribute to its prediction constraints. Our general model thus defines new problems and solutions, as in Section 5.1, and also offers insight into the complexity of numerous already studied online problems.

5.3 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 [13]. Alter-
natively, 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


A Undirected Incremental Cut

Theorem A.1 Undirected unweighted IC with demands is NP-hard for $k \geq 2$.

The proof of Theorem A.1 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.2 completes the proof.

![Figure 5: An unweighted MWC instance with $C = 3$ and its corresponding 2-level undirected IC instance.](image)

Lemma A.2 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 $S = (S_1, S_2) = (S, S \cup \{t_2, t\})$. Certainly $S$ is a feasible solution: $S_1 \subseteq S_2$, $S_1 \subseteq E_1$, 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

Lemma B.1 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 \( \ell_u \), remove it, because \( u \) must be covered by some other edge at or prior to level \( \ell_u \). Otherwise, replace \( e \) with \( e_v \) at level \( \ell_u \).
(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 \( \ell_v \), remove it. If \( e \) appears at or before level \( \ell_u \) keep \( e \) in our cover, but remove it from all levels prior to \( \ell_u \). Otherwise, \( e \) appears after level \( \ell_u \) but no later than level \( \ell_v \). In this case, replace \( e \) with \( e_v \) at level \( \ell_v \).

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. \( \square \)

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

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

\[ r_1 = \frac{v(S_1)}{v(S_1')} = \frac{s(a) + \sum_{a \in A \setminus A'} s(a)}{s(S)/2} = \frac{1+\phi}{2} \]

\[ r_2 = \frac{v(S_2)}{v(S_2')} = \frac{\phi \cdot s(S)/2}{\phi \cdot s(S)/2} = 1 \]

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