Algorithm Cover \((X, f, \omega)\)

If the set \(\{x \in X : \omega(x) = 0\}\) is a cover then return this set.
Select \(\omega\)-tight minimal-r-effective \(\delta : X \rightarrow R^+\)
Recursively call Cover \((X, f, \omega - \delta)\) to get a cover \(C\)

**Removal loop:**
for each \(x \in C\), if \(\delta(x) > 0\) and \(C \setminus \{x\}\) is a cover then \(C = C \setminus \{x\}\)

**Theorem 12.** Algorithm Cover is an \(r\)-approximation.

**Proof.** Define \(X^+ = \{x \in X : \omega(x) > 0\}\). Since \(\delta\) is \(\omega\)-tight, the size of \(X^+\) decreases at each recursive call. This implies that the total number of recursive calls is bounded by \(|X|\). We can now prove the theorem using induction on \(|X^+|\).

**Basis:** \(|X^+| = 0\), hence \(\omega(X) = 0\).

**Step:** We can consider the recursive call as the algorithm \(B\) in the Extended Local-Ratio theorem, theorem 8. By the induction hypothesis, this recursive call, Cover \((X, f, \omega - \delta)\), satisfies \(B\)'s requirement in the theorem, i.e., \((\omega - \delta)(X) \leq r \cdot \text{OPT}(\omega - \delta)\). Since \(\delta\) is minimal-r-effective, it remains to show that \(C\) is a minimal cover w.r.t \(\delta\). This follows from the last step of algorithm Cover.

**Acknowledgment**
We would like to thank Yishay Rabinovitch and Joseph Naor for helpful discussions, Avigail Orni for her careful reading and suggestions, and Yvonne Sagi for typing.

**References**

8 Generalized Steiner Forest and Related Problems

Goemans and Williamson [15] have recently presented a general approximation technique for covering cuts of vertices with edges. Their framework includes the Shortest Path, Minimum-cost Spanning Tree, Minimum-weight Perfect Matching, Traveling Salesman, and Generalized Steiner Forest problems. All these problems are called Constraint Forest Problems.

To simplify the presentation, let us choose the Generalized Steiner Forest problem. We are given a graph \( G = (V, E) \), a weight function \( \omega : E \rightarrow \mathbb{R}^+ \), and a collection of terminal sets \( T = \{T_1, T_2, \ldots, T_t\} \), each a subset of \( V \). A subset of edges \( C \subseteq E \) is called a Steiner-cover if, for each \( T_i \), and for every pair of terminals \( x, y \in T_i \), there exists a path in \( C \) connecting \( x \) to \( y \). The Generalized Steiner Forest problem is: given a triple \( (G, T, \omega) \), find a Steiner-cover \( C \) with minimum total weight \( \omega(C) \).

In order to get a \( \frac{1}{2} \)-effective weight function, let us define an edge-degree-weighted graph.

An edge-degree-weighted graph is a triple \( (G, T, \omega) \) s.t.

\[
\forall e \in E : \omega(e) = d_T(e) = |\{ x \in e : \exists y \in T_i \}|.
\]

**Theorem 11.** Every minimal cover in an edge-degree-weighted graph is a \( \frac{1}{2} \)-approximation.

**Proof.** The proof is an elementary exercise, see, e.g., [16]. \( \square \)

So now, given \( (G, T, \omega) \), we can easily compute a \( \frac{1}{2} \)-effective weight function \( \delta(e) = d_T(e) \cdot \epsilon \) for all \( e \in E \), and in order to guarantee termination, we select \( \epsilon = \min \{ \frac{\omega(e)}{d_T(e)} : d_T(e) \neq 0 \} \).

Now we can proceed as follows:

“Shrink” all pairs \( e = \{ x, y \} \) s.t. \( \omega - \delta(e) = 0 \).

Recursively, find a minimal \( \frac{1}{2} \)-approximation Steiner-cover, \( C \). Add all “shrunk” edges to the cover \( C \). In order to guarantee the minimality property, apply the following removal loop:

For each “shrunk” edge \( e \) do: if \( C \setminus \{ x \} \) is a Steiner-cover, delete \( e \) from \( C \).

9 A Generic \( r \)-Approximation Algorithm

We now present a generic \( r \)-approximation algorithm that can be used to generate all the deterministic algorithms presented in this paper.

The weight function \( \delta \) is called \( \omega \)-tight if \( \exists x \in X : \delta(x) = \omega(x) > 0 \).
The main idea behind Becker and Geiger’s algorithm is the weighted graph we call a degree-weighted graph. A degree-weighted graph is a pair \((G, \omega)\) s.t. each vertex \(v\) has a weight \(\omega(v) = d(v) > 1\).

**Theorem 9.** Every minimal FVS in a degree-weighted graph is a 2-approximation.

**Proof.** Let \(C\) be a minimal cover, and \(C^*\) an optimum cover. We need to prove \(\omega(C) \leq 2 \cdot \omega(C^*)\). It is enough to show that
\[
\sum_{x \in C} \omega(x) \leq 2 \cdot \sum_{x \in C^*} \omega(x)
\]
by definition, it is enough to show that
\[
\sum_{x \in C} d(x) \leq 2 \cdot \sum_{x \in C^*} d(x).
\]
This is proved in [9]. A simple proof also appears in [10].

**Corollary 2.** If \(G = (V, E)\) is a graph with \(\forall v \in V \ d(v) > 1\), and \(\delta : V \to \mathbb{R}^+\) satisfies \(\forall v \in V \ 0 \leq \delta(v) = \epsilon \cdot d(v) \leq \omega(v)\) for some \(\epsilon\), then \(\delta\) is 2-effective.

**Proof.** We need only show that for every minimal cover \(C, \delta(C) \leq 2 \cdot \text{OPT}(\delta)\). This is immediate from Theorem 9.

In order to guarantee termination, we choose \(\epsilon = \min_{v \in V} \frac{\omega(v)}{d(v)}\).

---

**Algorithm BeckerGeiger** \((G = (V, E), \omega)\)

- If \(G\) is a tree return \(\phi\)
- If \(\exists x \in V \ d(x) \leq 1\) return BeckerGeiger\((G \setminus \{x\}, \omega)\)
- If \(\exists x \in V \ \omega(x) = 0\) return \(\{x\} + \text{BeckerGeiger}(G \setminus \{x\}, \omega)\)
- Let \(\epsilon = \min_{v \in V} \frac{\omega(v)}{d(v)}\)
- Define \(\delta : \forall v \in V \ \delta(v) = \epsilon \cdot d(v)\)
- \(C = \text{BeckerGeiger}(G, \omega - \delta)\)
- **Removal loop:**
  for each \(x \in C\), if \(\delta(x) > 0\) and \(C \setminus \{x\}\) is a cover then \(C = C \setminus \{x\}\)

**Theorem 10.** Algorithm BeckerGeiger is a 2-approximation for FVC.

**Proof.** The algorithm terminates, since in each recursive call, at least one more vertex weight becomes zero. Therefore we can use induction. The base of induction is trivial. Let us assume inductively that the recursive call returns \(C\) s.t.
\[
(\omega - \delta)(C) \leq 2 \cdot \text{OPT}(\omega - \delta).
\]
Since, by Corollary 2, \(\delta\) is 2-effective, we can use the Extended Local-Ratio theorem to conclude that \(\omega(C) \leq 2 \cdot \text{OPT}(\omega)\).
Algorithm $A(X, f, \omega)$

Select a minimal-r-effective $\delta$
Call algorithm $B(X, f, \omega - \delta)$ to find a cover $C$

Removal loop:
for each $x \in C$, if $\delta(x) > 0$ and $C \setminus \{x\}$ is a cover then $C = C \setminus \{x\}$
Return $C$

Theorem 8. The Extended Local-Ratio Theorem
If $B$ returns a cover $C$ s.t. $(\omega - \delta)(C) \leq r \cdot \text{OPT}(\omega - \delta)$, then $\omega(C) \leq r \cdot \text{OPT}(\omega)$.

Proof. After the “removal loop”, $C$ is still a cover, but it is also minimal w.r.t. $\delta$, and therefore $\delta(C) \leq r \cdot \text{OPT}(\delta)$. Hence

$$
\omega(C) = (\omega - \delta)(C) + \delta(C) \\
\leq r \cdot \text{OPT}(\omega - \delta) + r \cdot \text{OPT}(\delta) \quad [\text{by the properties of } B \text{ and } \delta] \\
\leq r \cdot \text{OPT}(\omega) \quad [\text{by the Decomposition Observation}]
$$

7 The Feedback Vertex Set Problem

Let $G = (V, E)$ be a simple graph with weight function $\omega : V \to R^+$. A set $F \subseteq V$ is called a Feedback Vertex Set (FVS) if $G \setminus F$ is a forest, i.e., for every cycle $C \subseteq V$, $C \cap F \neq \emptyset$. The FVS problem is: given a graph $G = (V, E)$ and a weight function $\omega : V \to R^+$, find a FVS, $F$, with minimum total weight. The FVS problem is NP-hard [14]. Furthermore, any $r$-approximation for FVS implies an $r$-approximation with the same time complexity for the VC problem (each edge in the VC instance graph can be replaced by some cycle).

This implies that a 2-approximation is the best we can hope for, as long as the ratio for VC is not improved.

The first non-trivial approximation algorithm was given by Bar-Yehuda, Geiger, Naor and Roth [7]. They present a 4-approximation for unit weights and an $O(\log n)$-approximation for the general problem. Following their paper, Bafna, Berman and Fujito [2] found the first 2-approximation. They were also the first to generalize the local ratio theorem to deal with “paying” according to a minimal cover. Their approach has helped to deepen our understanding of approximations for covering problems.

Following their approach, Becker and Geiger [9] present a relatively simple 2-approximation. Following all these, Chudak, Goemans, Hochbaum and Williamson [10] explained all of these algorithms in terms of the primal-dual method. They also gave a new 2-approximation algorithm, which is a simplification of the Bafna et al. algorithm. Using the Extended Local Ratio theorem, we can further simplify the presentation and proof of all these three algorithms, [2], [10], [9]. Let us illustrate this for the Becker-Geiger algorithm.
Algorithm Random-Greedy \((G = (V, E), \omega)\)

\[
C = \{x : \omega(x) = 0\} \text{ and delete } C \text{ from } G
\]

While \(E \neq \emptyset\)

Let \(\tilde{\omega} = \frac{1}{\sum_{x \in V} \omega(x)} \sum_{x \in V} \omega(x)\)

Select at random \(x \in V\) with \(\text{Prob}(x) = \frac{\omega(x)}{\tilde{\omega}}\).

Remove \(x\) from \(G\) and add it to \(C\).

Return \(C\)

Let \(C^*\) be an optimum cover,

\[
\text{Exp}[\text{OPT}(\delta)] = \text{Exp}[\delta(C^*)] = \sum_{x \in C^*} p(x) \cdot \omega(x) = \sum_{x \in C^*} d(x) \cdot \tilde{\omega} \geq |E| \cdot \tilde{\omega}
\]

\[
\text{Exp}[\delta(V)] = \sum_{x \in V} p(x) \cdot \omega(x) = \sum_{x \in V} d(x) \cdot \tilde{\omega} \leq \Delta \cdot |E| \cdot \tilde{\omega}
\]

this implies:

**Theorem 7.** Random-Greedy is a randomized-\(\Delta\)-approximation for the HVC problem.

**Corollary 1.** Random-Greedy is a randomized-2-approximation for VC.

### 6 Pay Only for Minimal Covers

Following Bafna, Berman and Fujito [2], we extend the Local-Ratio theorem. This will be shown to be useful in the following sections.

In the previous analyses, we used \(\delta(V)\) as an upper bound for the payment in a step. This may be too much. If we knew the final cover \(C\) eventually found by the algorithm, we would know that \(\delta(C)\) is the exact payment.

In practice, we can restrict the final cover \(C\) to be a minimal cover. Defined formally:

**Definition** A cover \(C\) is a minimal cover if \(\forall x \in C \setminus \{x\}\) is not a cover. A cover \(C\) is a minimal cover w.r.t a weight function \(\omega\), if \(\forall x \in C\) and \(\omega(x) \geq \delta\) \(C \setminus \{x\}\) is not a cover.

If we know that the final cover \(C\) is minimal w.r.t. \(\delta\), we can bound the payment \(\Delta \cdot \text{COST}\) by \(\max \{\delta(C) | C\text{ is minimal w.r.t } \delta\}\).

Define \(\delta : X \rightarrow R^+\) to be minimal-\(r\)-effective if

1. \(\forall x \in X\) \(0 \leq \delta(X) \leq \omega(x)\)
2. \(\delta(C) \leq r \cdot \text{OPT}(\omega)\) for all \(C\) that are minimal w.r.t. \(\delta\).

Consider the following general algorithm \(A\):
5.1 Pitt’s Randomized-2-Approximation Algorithm

Algorithm Pitt \( G = (V, E), \omega \) 

\[
C = \{x : \omega(x) = 0\} \text{ and delete } C \text{ from } G \\
\text{While } E \neq \emptyset \\
\quad \text{Select } e \in E \\
\quad \text{Let } \omega = \frac{1}{\sum_{e \in E} \omega(e)} \\
\quad \text{Select at random } x \in e \text{ with } \text{Prob}(x) = \frac{\omega(x)}{\omega(e)} \\
\quad \text{Remove } x \text{ from } G \text{ and add it to } C \\
\text{Return } C
\]

Pitt [20] uses the primal-dual approach to prove that his algorithm is a randomized-2-approximation algorithm for Vertex Cover. Using our technique, we get:

**Theorem 6.** Pitt’s algorithm is a randomized-\(\Delta_v\)-approximation for HVC.

**Proof.** To prove correctness, we use the Randomized Local-Ratio theorem. We can regard the selection of a vertex \(x\) and its removal as if we defined \(d(x) = \omega(x)\), and \(\forall y \neq x, \delta(y) = 0\). We show that in each iteration, the distribution is \(\Delta_v\)-effective. For this, we need to show that \(Exp[\delta(V)] \leq r \cdot Exp[OPT(\delta)]\).

Let \(e\) be the edge chosen in the first iteration, and let \(C^*\) be an optimum cover.

\[
Exp[OPT(\delta)] = Exp[\delta(C^*)] = \sum_{x \in e \cap C^*} p(x) \cdot \omega(x)
\]

\[
= \sum_{x \in e \cap C^*} \frac{\omega(x)}{\omega(e)} \cdot \omega(x) = \omega \cdot |e \cap C^*| \geq \omega
\]

\[
Exp[\delta(V)] = \sum_{x \in V} p(x) \cdot \omega(x) = \sum_{x \in e} \frac{\omega(x)}{\omega(e)} \cdot \omega(x) = \omega \cdot |e| \leq \omega \cdot \Delta_v
\]

\[\square\]

5.2 Our Modified Greedy Algorithm

The greedy algorithm for set cover [11] is a \(ln\Delta_v\)-approximation. Algorithm BarEven is a \(\Delta_v\)-approximation. In some families of problems (e.g., VC), \(\Delta_v\) is preferable to \(ln\Delta_v\). In these cases, we would like to have a ratio of \(\Delta_v\), but we may also prefer to use the “greedy” heuristic. Can we have the best of both worlds? As we have seen, this is possible: Clarkson’s modified greedy algorithms for VC can easily be modified for HVC, using the Local-Ratio theorem. But, even for unit-weight Vertex Cover, it may have a problem with robustness, due to error accumulation. The following randomized greedy modification does not “reduce weights”, but rather deletes vertices:

\[
\text{Return } C
\]
maximum vertex degree is defined by \( \Delta_v = \max\{d(v) : v \in V\} \). The degree of an edge \( e \in E \) is defined by \( d(e) = |e| \). The maximum edge degree is defined by \( \Delta_e = \max\{|e| : e \in E\} \). A set \( C \subseteq V \) is called a vertex cover of \( G = (V, E) \) if every edge \( e \in E \) has at least one endpoint in \( C \) (i.e., \( e \cap C \neq \emptyset \)). The Hypergraph Vertex Cover problem (HVC) is: given \( G = (V, E) \), and for each \( v \in V \) a weight \( \omega(v) \geq 0 \), find a cover with minimum total weight.

This generalization of the VC problem is exactly the Set Covering problem (SC).

Algorithm BarEven is written in hypergraph terminology, and therefore approximates HVC as well.

**Theorem 4.** Algorithm BarEven is a \( \Delta_e \)-approximation for HVC.

*Proof.* For \( e \in E \), \( \epsilon = \min\{\omega(v) : v \in e\} \). Define \( \delta(v) = \epsilon \) for all \( v \in e \) and 0 otherwise.

Since \( \text{OPT}(\delta) = \epsilon \) and \( \text{OPT}(V) = \delta(e) = |e| \cdot \epsilon \leq \Delta_e \cdot \epsilon \), this implies that \( \delta \) is \( \Delta_e \)-effective.

\[ \square \]

### 5 Randomized \( r \)-approximations

Let us select \( \delta : X \rightarrow R^+ \) from some random distribution. Such a random distribution is called \( r \)-effective if

1. \( \forall x \in X \quad 0 \leq \delta(X) \leq \omega(x) \) and
2. \( \text{Exp}[\delta(X)] \leq r \cdot \text{Opt}[\text{OPT}(\delta)] \)

Intuitively: the randomized rate of payment is no more than \( r \) times the randomized rate of the optimum reduction.

Now consider the following general algorithm:

**Algorithm** \( A(X, f, \omega) \)

Select \( \delta : X \rightarrow R^+ \) from some \( r \)-effective random distribution
Call algorithm \( B(X, f, \omega - \delta) \) to find a cover \( C \)
Return \( C \)

**Theorem 5.** The Randomized Local-Ratio Theorem:
If \( B \) returns a cover \( C \) s.t. \( \text{Exp}[\omega(C)] \leq r \cdot \text{Opt}[\text{OPT}(\omega - \delta)] \),
then \( \text{Exp}[\omega(C)] \leq r \cdot \text{Opt}[\text{OPT}(\omega)] \).

*Proof.* \( \text{Exp}[\omega(C)] = \text{Exp}[\omega(C)] + \text{Exp}[\delta(C)] \leq r \cdot \text{Exp}[\text{OPT}(\omega - \delta)] + r \cdot \text{Exp}[\text{OPT}(\delta)] \) [from the properties of \( B \) and of the distribution ]

\[ \leq r \cdot \text{Opt}[\text{OPT}(\omega - \delta)] + \text{Opt}[\delta] \]

\[ \leq r \cdot \text{Opt}[\text{OPT}(\omega)] \] [by the Decomposition Observation.]

\[ \square \]

An algorithm \( A \) that produces a cover \( C \) satisfying \( \text{Exp}[\omega(C)] \leq r \cdot \text{OPT}(\omega) \)
is called a randomized-\( r \)-approximation.
Algorithm Clarkson \((G = (V, E), \omega)\)

\[
C = \phi \\
\text{While } E \neq \phi \\
\quad \text{Let } x \in V \text{ with minimum } \epsilon = \frac{\omega(x)}{d(x)} \\
\quad \text{For each } y \in \Gamma(x) \text{ (the neighbors of } x) \\
\quad \quad \omega(y) = \omega(y) - \epsilon \\
\quad \text{Add } x \text{ to } C \text{ and remove it from } G \\
\text{Return } C
\]

**Theorem 3.** Clarkson’s Algorithm is a 2-approximation.

**Proof.** The \(\delta\) that the algorithm chooses is \(\delta(x) = \omega(x), \delta(y) = \frac{\omega(x)}{d(x)}\) for every \(y \in \Gamma(x)\), and 0 for all others. We need to show that \(\delta\) is 2-effective, i.e. (1) \(\forall v \in V \delta(v) \leq \omega(v)\) (2) \(\delta(V) \leq 2 \cdot \OPT(\delta)\).

1. For \(v = x\) we defined \(\delta(x) = \omega(x)\). For \(v \in \Gamma(x)\), we defined \(\delta(v) = \frac{\omega(x)}{d(v)} \leq \frac{\omega(x)}{d(x)} \leq \omega(x)\). For all others, \(\delta(v) = 0 \leq \omega(v)\).
2. \(\delta(V) = \delta(x) + \delta(\Gamma(x)) + \delta(V - \{x\} - \Gamma(x)) = \omega(x) + \omega(x) + 0 = 2 \cdot \omega(x) = 2 \cdot \OPT(\delta)\)

The rest of the proof is by induction, using the Local Ratio Theorem. \(\square\)

### 3.3 Subgraph Removal

Suppose we are given a family of graphs for which we wish to develop a 1.5-approximation. Getting rid of triangles is “free”. To get rid of a triangle, say \(x, y, z\), with weights \(\epsilon = \omega(x) \leq \omega(y) \leq \omega(z)\), define \(\delta(x) = \delta(y) = \delta(z) = \epsilon\), and \(\forall v \notin \{x, y, z\} \delta(v) = 0\). Obviously, \(\OPT(\delta) = 2\epsilon\), while \(\delta(\{x, y, z\}) = 3\epsilon\), and therefore \(\delta\) is 1.5-effective.

This was helpful, for example, in developing an efficient 1.5-approximation for planar graphs, see [4].

The \((2 - \log \log n / (21 \log n))\)-approximation algorithm [4] uses this approach to “get rid” of odd cycles of length \(\leq 2k - 1\), where \(k = \log n / \log \log n\). Let \(C \subseteq V\) be a cycle of length \(2k - 1\). Generalizing the idea for triangles, let \(\epsilon = \min \{\omega(v) : v \in C\}\), and define \(\delta(v) = \epsilon\) if \(v \in C\), and 0 otherwise. In this case, \(\delta(C) / \OPT(\delta) \leq (2k - 1) / k = 2 - \frac{1}{k}\).

Bar-Yehuda and Even’s linear time algorithm for VC can also be viewed as a subgraph removal algorithm, where the subgraph removed each time is an edge.

### 4 Vertex Cover in Hypergraphs

Let \(G = (V, E)\) be a simple\(^2\) hypergraph with vertices \(V\) and edges \(E \subseteq 2^V\). The **degree of a vertex** \(v \in V\) is defined by \(d(v) = |\{e \in E : v \in e\}|\). The

\(^2\) A hypergraph is called simple if no edge is a subset of another edge.
Nemhauser and Trotter [18] developed a local optima algorithm that (trivially) implies a 2-approximation.

Until today, the best ratio known is 2. The first linear time algorithm was found by Bar-Yehuda and Even [3]. Their proof uses the primal-dual approach. It took a few more years to find a different kind of proof – the local-ratio theorem [4].

### 3.1 Bar-Yehuda & Even’s 2-Approximation Algorithm

The main idea is as follows: let \( \{x, y\} \in E \), and let \( \epsilon = \min(\omega(x), \omega(y)) \). Reducing \( \omega(x) \) and \( \omega(y) \) by \( \epsilon \) has \( \Delta \text{COST} = 2\epsilon \) while \( \Delta \text{OPT} \geq \epsilon \), since one of the two vertices \( x, y \) participates in any optimum. So we pay \( 2\epsilon \) in order to reduce the total optimum by at least \( \epsilon \), repeating this process until the optimum reaches zero. The total cost will be no more than twice the optimum.

Therefore, the 2-effective \( \delta \) selected is

\[
\delta(v) = \begin{cases} 
\epsilon & v \in \{x, y\} \\
0 & \text{else}
\end{cases}
\]

```plaintext
Algorithm BarEven(G = (V, E), \omega)

For each \( \epsilon \in E \)
   Let \( \epsilon = \min\{\omega(v) | v \in \epsilon\} \)
   For each \( v \in \epsilon \)
      \( \omega(v) = \omega(v) - \epsilon \)
   Return \( C = \{v : \omega(v) = 0\} \)
```

**Theorem 2.** Algorithm BarEven is a 2-approximation.

*Proof.* By induction on \( |E| \).

The first step uses \( \delta \), which is 2-effective, since \( \text{OPT}(\delta) = \epsilon \) and \( \delta(V) = 2\epsilon \). The remaining steps perform a 2-approximation for \((G, \omega - \delta)\), by the induction hypothesis, and the Local Ratio Theorem. \( \square \)

For primal-dual proof see [3]. For first local-ratio proof, see [4].

### 3.2 Clarkson’s Greedy Modification

Clarkson [12] uses the primal-dual approach to get the following 2-approximation algorithm.
Let us be generous and pay for all weight reductions we make, i.e., define \( \Delta \text{COST} \) to be \( \Delta \text{COST} = \sum_{x \in X} \delta(x) = \delta(X) \). Therefore, what we need in order to get an \( r \)-approximation is
\[
\delta(X) \leq r \cdot \text{OPT}(\delta)
\]
or, in the terminology of the title: we are willing to pay no more than \( r \) times the gain in optimum reduction.

Let us now formalize the technique. Let \( (X, f, \omega) \) be a covering problem triple. The function \( \delta \) is called \( r \)-effective if:

1. \( \forall x \in X \quad 0 \leq \delta(x) \leq \omega(x) \)
2. \( \delta(X) \leq r \cdot \text{OPT}(\delta) \).

Now, consider the following general algorithm:

**Algorithm** \( A(X, f, \omega) \):
- Select \( r \)-effective \( \delta \)
- Call algorithm \( B(X, f, \omega - \delta) \) to find a cover \( C \)
- Return \( C \)

**Theorem 1. The Local-Ratio Theorem:**
If \( B \) returns a cover \( C \) s.t. \( (\omega - \delta)(C) \leq r \cdot \text{OPT}(\omega - \delta) \) then \( \omega(C) \leq r \cdot \text{OPT}(\omega) \)

**Proof.** \( \omega(C) = \)
- \( (\omega - \delta)(C) + \delta(C) \) [linearity of \( \omega \) and \( \delta \)]
- \( (\omega - \delta)(C) + \delta(X) \) [\( \delta(C) \leq \delta(X) \)]
- \( \leq r \cdot \text{OPT}(\omega - \delta) + r \cdot \text{OPT}(\delta) \) [\( B \)'s property and the \( r \)-effectiveness of \( \delta \)]
- \( \leq r \cdot \text{OPT}(\omega) \) [Decomposition Observation]

This is illustrated in the next section by a sequence of \( 2 \)-approximation algorithms for the Vertex Cover problem.

### 3 The Vertex Cover Problem

Let \( G = (V, E) \) be a simple graph with vertices \( V \) and edges \( E \subseteq V^2 \). The degree of a vertex \( v \in V \) is defined by \( d(v) = \{ e \in E : v \in e \} \). The maximum vertex degree is defined by \( \Delta = \max \{ d(v) : v \in V \} \). A set \( C \subseteq V \) is called a vertex cover of \( G = (V, E) \) if every edge has at least one endpoint in \( C \), i.e., \( \forall e \in E \cap C \neq \emptyset \).

The Vertex Cover problem (VC) is: given a simple graph \( G = (V, E) \) and a weight \( \omega(v) > 0 \) for each \( v \in V \), find a vertex cover \( C \) with minimum total weight. The Vertex Cover problem (VC) is NP-hard even for planar cubic graphs with unit weight [14]. For unit-weight Vertex Cover, Gavril (see [14]) suggested a linear time 2-approximation algorithm. For the general Vertex Cover problem,
The objective of the problem is to find a cover $C^* \subseteq X$ satisfying
\[ \omega(C^*) = \min \{ \omega(C) | C \subseteq X \text{ and } f(C) = 1 \}. \]

Such a cover is called an **optimum cover**.

A cover $C \subseteq X$ is called an $r$-**approximation** if $\omega(C) \leq r \cdot \omega(C^*)$, where $C^*$ is an optimum cover. An algorithm $A$ is called an $r$-**approximation** for a family of covering problems if, for every input $(X, f, \omega)$ in the family, $A$ returns a cover which is an $r$-approximation.

### 1.2 Overview

This paper describes a unified approach for achieving $r$-approximations for covering problems. In Section 2 we describe the general idea of the technique, formalized by the Local-Ratio theorem. In Section 3 we show some applications of this technique to developing algorithms for the Vertex Cover problem. In Section 4 we apply the technique to Set Covering. In Section 5 we present an extension for dealing with randomized approximations. In Section 6 we describe an enhancement of the technique which allows us to deal with some recent approximations. In Section 7 we use the enhancement to describe and prove a 2-approximation algorithm for the Feedback Vertex Set problem, in Section 8 we use it for a 2-approximation for the Generalized Steiner Forest problem, and in Section 9 we present a generic $r$-approximation algorithm which can generate all deterministic algorithms presented in this paper.

### 2 First Technique: Pay for Every Weight Reduction

Given a triple $(X, f, \omega)$, our method for achieving an approximation cover is by manipulating the weight function $\omega$. Denote the optimum cover of $(X, f, \omega)$ by $\text{OPT}(\omega)$. The following is a fundamental observation for our technique:

**The Decomposition Observation:** For every two weight functions $\omega_1, \omega_2$,
\[ \text{OPT}(\omega_1) + \text{OPT}(\omega_2) \leq \text{OPT}(\omega_1 + \omega_2) \]

We select some function $\delta : X \rightarrow \mathbb{R}^+$, s.t. $\forall x \in X \ 0 \leq \delta(x) \leq \omega(x)$

The function $\delta$ is used to reduce the weights, giving a new instance $(X, f, \omega - \delta)$. The value of $\text{OPT}(\omega - \delta)$ may be smaller than $\text{OPT}(\omega)$, and we denote their difference by $\Delta \text{OPT} = \text{OPT}(\omega) - \text{OPT}(\omega - \delta)$. By the Decomposition Observation, $\Delta \text{OPT} \geq \text{OPT}(\delta)$.

This means that by reducing the weights by $\delta$, we reduce the total optimum by at least $\text{OPT}(\delta)$. How much are we willing to pay for this reduction in the optimum? Since our final objective is an $r$-approximation, we would like to pay no more than $r \cdot \text{OPT}(\delta)$. 

\[ \Delta \text{OPT} \geq \text{OPT}(\delta) \leq r \cdot \text{OPT}(\delta). \]
The common characteristic that we find in algorithms for covering problems is the use of vertex weight reductions (or vertex deletion, which is a special case). We are interested in viewing algorithms from the viewpoint of the Local-Ratio theorem [4], and its extensions by Bafna, Berman and Fujito [2].

Following the work of Bafna et al., several studies have shown unified approaches for various families of covering problems. Fujito [13] gave a generic algorithm for node deletion. Chudak, Goemans, Hochbaum and Williamson [10] united algorithms for the Feedback Vertex Set problem presented by Bafna et al., [2] and by Becker and Geiger [9], using the primal-dual principle. Agrawal, Klein and Ravi [1] dealt with the Generalized Steiner Forest problem. Following their work, a couple of generalizations/unifications have recently been done, one by Williamson, Goemans, Mihail, and Vazirani [21] and another by Goemans and Williamson [15].

In this work, we try to combine all of these results into a unified definition for covering problems, and present a new local-ratio principle, which explains, with relative ease, the correctness of all the above algorithms. Like the others, we use reductions of the weights in the graph, but we view this in the following way:

We try to measure the “effectiveness” of the reduction we are making; as a result of the reduction, there may be a change in the optimum. The reduction incurs a cost, and we would like the payment to be bounded by a constant $r$ times the change in the problem’s optimum.

This principle of “effectiveness” can also be viewed in terms of expected values, when the reduction we make is chosen according to some random distribution. This yields an extension of the local-ratio principle in another direction; in this extension, we are interested in bounding the expected change in the optimum as related to the expected cost.

This approach gives us a simple proof for Pitt’s randomized approximation for Vertex Cover.

In addition, we get a new and interesting result: while a deterministic greedy algorithm [11] does not have a constant performance ratio, even for Vertex Cover, a randomized greedy algorithm has a performance ratio of 2.

### 1.1 Definitions

We are interested in defining a covering problem in such a way that it will contain the following problems: Vertex Cover, Set Cover, Feedback Vertex Set, Generalized Steiner Forest, Min 2-Sat, Min Cut, etc.

For a general covering problem, the input is a triple $(X, f : 2^X \to [0, 1], \omega : X \to R^+)\), where $X$ is a finite set, $f$ is monotone, i.e., $A \subseteq B \implies f(A) \leq f(B)$, and $f(X) = 1$. For a set $C \subseteq X$, we define $\omega(C) = \sum_{x \in C} \omega(x)$, which we call the weight of $C$. We say that $C \subseteq X$ is a cover if $f(C) = 1$.

\footnote{For recent applications see [5] [6] [8].}
One for the Price of Two: A Unified Approach for Approximating Covering Problems *  
(Second edition) **

Reuven Bar-Yehuda

Computer Science Dept, Technion IIT, Haifa 32000, Israel  
reuven@cs.technion.ac.il,
WWW home page: http://cs.technion.ac.il/~reuven

Abstract. We present a simple and unified approach for developing and analyzing approximation algorithms for covering problems. We illustrate this on approximation algorithms for the following problems: Vertex Cover, Set Cover, Feedback Vertex Set, Generalized Steiner Forest and related problems. The main idea can be phrased as follows: iteratively, pay two dollars (at most) to reduce the total optimum by one dollar (at least), so the rate of payment is no more than twice the rate of the optimum reduction. This implies a total payment (i.e., approximation cost) is at most twice the optimum cost.

Our main contribution is based on a formal definition for covering problems, which includes all the above fundamental problems and others. We further extend the Bahna, Berman and Fujito Local-Ratio theorem. This extension eventually yields a short generic $r$-approximation algorithm which can generate most known approximation algorithms for most covering problems.

Another extension of the Local-Ratio theorem to randomized algorithms gives a simple proof of Pitt’s randomized approximation for Vertex Cover. Using this approach, we develop a modified greedy algorithm, which for Vertex Cover, gives an expected performance ratio $\leq 2$.

Keywords: Approximation Algorithm, Local Ratio, Covering Problems, Vertex Cover, Set Cover, Feedback Vertex Set, Generalized Steiner Forest, Randomized Approximations.

1 Introduction

Most algorithms for covering problems can be viewed according to two different approaches: the primal-dual principle, and the Local-Ratio principle (see, e.g.,

* This research was supported by the fund for the promotion of research at the Technion.

** To appear in APPROX98. This is a modified version of: Computer Science Department Technion - IIT, Israel TR CS0920, 1997