A LOCAL-RATIO THEOREM FOR APPROXIMATING THE WEIGHTED VERTEX COVER PROBLEM

by

R. Bar-Yehuda and S. Even

Technical Report #260

January 1983

* Supported by the Fund for the promotion of Research at the Technion
ABSTRACT

A local-ratio theorem for approximating the weighted vertex-cover problem is presented. It consists of reducing the weights of vertices in certain subgraphs and has the effect of local-approximation.

Putting together the Nemhauser-Trotter local optimization algorithm and the local-ratio theorem yields several new approximation techniques which improve known results from time-complexity, simplicity and performance-ratio points of view.

The main approximation algorithm guarantees a ratio of

\[ 2 \cdot \frac{1}{k} \]

where \( k \) is the smallest integer s.t. \((2k - 1)^k \geq n\) (hence

\[ \text{ratio} \leq 2 \cdot \frac{\log \log n}{\log n}. \]

This is an improvement over the currently known ratios especially for a "practical" number of vertices (e.g., for graphs which have less than 2400, 60000, 1.012 vertices the ratio is bounded by 1.75, 1.8, 1.9 respectively).

* All log bases in this paper are 2.
1. INTRODUCTION

A **Vertex Cover** of a graph is a subset of vertices such that each edge has at least one endpoint in the subset. The **Weighted Vertex Cover Problem** (WVC) is defined as follows: Given a simple graph $G(V,E)$ and a weight function $w: V \rightarrow R^+$, find a cover of minimum total weight. WVC is known to be NP-Hard, even if all weights are 1 [K72] and the graph is planar [GJ79]. Therefore, it is natural to look for efficient approximation algorithms.

Let $A$ be an approximation algorithm. For graph $G$ with weight functions $w$, let $C_A, C^*$ be the cover $A$ produces and an optimum cover, respectively. Define

$$R_A(G,w) = \frac{w(C_A)}{w(C^*)}$$

and let the performance ratio $r_A(n)$ be

$$r_A(n) = \sup \{ R_A(G,w) \mid G=(V,E) \text{ where } n=|V| \}.$$  

Many approximation algorithms with performance ratio $\leq 2$ have been suggested; see, for example Table 1. No polynomial-time approximation algorithm $A$ is known for which $r_A(n) \leq 2 - \varepsilon$, where $\varepsilon > 0$ and fixed. Several approximation algorithms are known for which $R_A(G,w) \leq 2 - \varepsilon(G)$, where $\varepsilon$ depends on $G$; e.g. $\varepsilon(G) = \frac{2}{\Delta(G)}$ where $\Delta(G)$ is the maximum degree of the vertices of $G$.

In Section 2, a new theorem, the 'Local-Ratio Theorem' is presented and proved.

In Section 3 we review the Nemhauser and Trotter [NT75] local optimization algorithm (NT) and note several properties of this technique which are useful in the following section.

In Section 4, we present two approximation algorithms in which the NT algorithm and the local-ratio theorem are shown to be useful. The first algorithm COVER2 satisfies $r_{COVER2}(n) \leq 2 - \frac{1}{\sqrt{n}}$ for general graphs, while for planar graphs $r_{COVER2}(n) \leq 1.5$ and its time complexity is $O(n^2 \log n)$. Hochbaum [HB81] obtained the same performance ratio, but we manage to avoid the time complexity of 4-coloring.
For our main algorithm, COVER3, we prove that \( \tau_{\text{COVER3}}(n) \leq 1 - \frac{1}{k} \), where \( k \) is the least integer s.t. \((2k - 1)^k \geq n\). A similar result, but only for unweighted graphs, has been obtained, independently by Monien and Speckenmeyer [MS83].

### Table 1: Summary of Results

(\textit{approximation algorithms for WVC with } \( G = (V, E) \))

<table>
<thead>
<tr>
<th>Reference</th>
<th>Performance ratio ( \leq )</th>
<th>Complexity for weighted</th>
<th>Complexity for unweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Gv74]</td>
<td>2</td>
<td>not applicable</td>
<td>( E )</td>
</tr>
<tr>
<td>[H82]</td>
<td>2</td>
<td>( V^3 )</td>
<td></td>
</tr>
<tr>
<td>[BE81]</td>
<td>2</td>
<td>( E )</td>
<td></td>
</tr>
<tr>
<td>[H81]</td>
<td>( 2 - \frac{2}{\Delta} )</td>
<td>&quot;NT&quot;</td>
<td>&quot;NT&quot;</td>
</tr>
<tr>
<td>this paper</td>
<td>( 2 - \frac{2}{\sqrt{V}} )</td>
<td>&quot;NT&quot;</td>
<td>&quot;NT&quot;</td>
</tr>
<tr>
<td>this paper</td>
<td>( 2 - \frac{\log \log V}{2 \log V} )</td>
<td>&quot;NT&quot;</td>
<td>( EV )</td>
</tr>
</tbody>
</table>

(For planar graphs)

<table>
<thead>
<tr>
<th>Reference</th>
<th>Performance ratio ( \leq )</th>
<th>Complexity for weighted</th>
<th>Complexity for unweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>[H81]</td>
<td>1.6</td>
<td>&quot;NT&quot;</td>
<td>( V^{1.5} )</td>
</tr>
<tr>
<td>[H81]</td>
<td>1.5</td>
<td>&quot;NT&quot; + &quot;4-COLORING&quot;</td>
<td>&quot;NT&quot; + &quot;4-COLORING&quot;</td>
</tr>
<tr>
<td>this paper</td>
<td>1.5</td>
<td>&quot;NT&quot;</td>
<td>&quot;NT&quot;</td>
</tr>
<tr>
<td>[BE82]</td>
<td>( \frac{2}{3} )</td>
<td>not applicable</td>
<td>( V )</td>
</tr>
<tr>
<td>[BE82]</td>
<td>( 1 + \varepsilon )</td>
<td>not applicable</td>
<td>( V \log V )</td>
</tr>
</tbody>
</table>

### Table 2: "NT"'s Complexity

(\textit{the same as MAX FLOW})

<table>
<thead>
<tr>
<th>( G = (V, E) )</th>
<th>weighted</th>
<th>unweighted</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>( E^{2/3} V^{2/3} ) or ( E V \log V )</td>
<td>( E \sqrt{V} )</td>
</tr>
<tr>
<td>Planar</td>
<td>( V \log V )</td>
<td>( V^{1.5} )</td>
</tr>
</tbody>
</table>
2. THE LOCAL-RATIO THEOREM

In a previous paper [BE82] we presented a local approximation technique for the vertex cover problem of unweighted graphs. In this section we present a local approximation technique for the vertex cover problem of weighted graphs. First, we present the following lemma:

**Lemma:** Let $G(V,E)$ be a graph and $w, w_1$ and $w_2$ be weight functions on $V$, s.t. for every $v \in V$: $w(v) = w_1(v) + w_2(v)$. Let $C^*$, $C_1^*$ and $C_2^*$ be optimum covers of $G$ with respect to $w, w_1$ and $w_2$. It follows that: $w(C^*) = w_1(C_1^*) + w_2(C_2^*)$.

**Proof:**

\[ w(C^*) = \sum_{v \in C^*} w(v) \]

\[ \geq \sum_{v \in C^*} (w_1(v) + w_2(v)) \]

\[ = w_1(C^*) + w_2(C^*) \]

\[ \leq w_1(C_1^*) + w_2(C_2^*) \]

[by the optimality of $C_1^*$ and $C_2^*$].

Q.E.D.

Let $\tilde{G}$ be an unweighted graph of $\overline{n}$ vertices, whose optimum cover contains $\overline{\delta^*}$ vertices. Define $\overline{\delta} = \overline{n} / \overline{\delta^*}$. Let $A$ be an approximation algorithm for WVC and let $LOCAL_\overline{G}$ be the following algorithm:

**Algorithm** $LOCAL_\overline{G}$

**Input:** Graph $G(V,E)$, with weight function $w$.

**Phase 1:** Choose a subgraph $\tilde{G}(\tilde{V}, \tilde{E})$ of $G$ which is isomorphic to $\tilde{G}$.

Choose $0 \leq \delta \leq \min \{ w(x) \mid x \in \tilde{V} \}$

Define $w_\overline{\delta}(x) = \begin{cases} w(x) - \delta & \text{if } x \in \tilde{V} \\ w(x) & \text{else.} \end{cases}$
Phase 2: Call \( A(G,w_0) \) to get \( C_0 \).

Output: \( C = C_0 \).

**The Local-Ratio Theorem:** \( R_{\text{LOCAL}_0}(G,w) \leq \max \{ r, R_A(G,w_0) \} \).

**Proof:** Let \( c^* \) and \( c_0^* \) be the weights of the optimum covers of \( G \) with respect to \( w \) and \( w_0 \), and let \( r = \max \{ r, R_A(G,w_0) \} \), then

\[
w(C) \leq w_0(C) + \delta \cdot m \quad \text{[by definitions]}
\]

\[
\leq R_A(G,w_0) \cdot c_0^* + r \cdot \delta \cdot c^* \quad \text{[by definitions]}
\]

\[
\leq r \cdot (c_0^* + \delta c^*) \quad \text{[by r's definition]}
\]

\[
\leq r \cdot c^* \quad \text{[by the lemma]}
\]

Q.E.D.

Let us consider now a generalization of the Local-Ratio Theorem. Let \( \Gamma \) be a finite family of graphs, and \( r_\Gamma = \max \{ r | \exists G \in \Gamma \} \).

We denote by \( G(U), U \subseteq V \), the subgraph of \( G(V,E) \) induced by \( U \).

**Algorithm ** \( \text{LOCAL}_\Gamma \)

**Input:** \( G(V,E), w. \)

**Phase 0:** For every \( x \in V \) do \( u_0(x) \leftarrow w(x) \) end

**Phase 1:** For every \( \tilde{G}(\tilde{V}, \tilde{E}) \), subgraph of \( G \) which is isomorphic to some \( G \in \Gamma \), do

\[
\delta \leftarrow \min \{ w_0(x) | x \in \tilde{V} \}, \quad \text{do}
\]

\[
\text{For every } x \in \tilde{V} \text{ do } u_0(x) \leftarrow w_0(x) - \delta \text{ end}
\]

end

**Phase 2:** \( C_1 \leftarrow \{ x | u_0(x) = 0 \} \).

\( V_1 \leftarrow V - C_1 \).

Call \( A(G(V_1), w_0) \) to get \( C_2 \).

Output: \( C = C_1 \cup C_2 \).
The Generalized Local-Ratio Theorem: \( R_{\text{LOCAL}_i}(G,w) \leq \max \{ r_\Gamma | R_\Gamma (G(V_i), w_i) \} \).

**Proof:** By induction on \( i \), the number of iterations of Phase 1, during which \( \delta > 0 \).

For \( i = 0 \) the claim is trivial.

Suppose the claim holds for \( i \), and for some \( G,w \) Phase 1 runs \( i+1 \) iterations during which \( \delta > 0 \). Let \( \mathcal{G} \in \Gamma \) be the graph used in the first iteration during which \( \delta > 0 \). Observe that we can view the running of \( \text{LOCAL}_\Gamma \) as an application of \( \text{LOCAL}_\mathcal{G} \) where \( A \) (in Phase 2 of \( \text{LOCAL}_\mathcal{G} \)) is replaced by the remaining part of \( \text{LOCAL}_\Gamma \). The inductive step is now an immediate consequence of the Local-Ratio Theorem.

Q.E.D.

In the applications of \( \text{LOCAL}_\Gamma \) we shall refer to \( r_\Gamma \) as the local-ratio of Phase 1.

Let us demonstrate a simple application of the Generalized Local-Ratio Theorem.

**Algorithm COVER**

**Input:** \( G(V,E), w \).

**Phase 1:** For every \( e \in E \) do

Let \( \delta = \min \{ w(x) | x \in e \} \).

For every \( x \in e \) do \( w(x) = w(x) - \delta \) end

end

**Output:** \( C = \{ x | w(x) = 0 \} \).

This linear time algorithm is essentially the one we described in [BE81]. Its performance ratio \( \leq 2 \), by the Generalized Local-Ratio Theorem, with \( \Gamma \) consisting of a single graph \( \mathcal{G} \) which consists of a single edge.

**Note:**

Let HWVC be the following problem. Given a hypergraph \( G = (V,E) \) with weight function \( w: V \rightarrow \mathbb{R}^+ \), find a set \( C \subseteq V \) of minimum total weight s.t. for every \( e \in E \),

\[ w(C \cup e) = w(C) + w(e) \text{ or } w(C \cup e) = w(C) - w(e) \]

In [BE81] we used a global rather than a local point of view.
The Local-Ratio Theorem and its generalization hold also for HWVC. Algorithm COVER1 can be applied directly to HWVC with \( \tau_{\text{COVER1}} \leq \Delta \) [where \( \Delta \) is the maximum edge-degree (or cardinality) in \( G \)]. Its running time is linear in the length of the problem's input \( (\sum |e|) \). HWVC is an extension of the Weighted-Set-Cover Problem which itself is an extension of WVC \([\Delta \leq 2]\). Even though we get performance-ratio \( \leq \Delta \) in linear time, we suspect that for any fixed \( \Delta \), there is no polynomial time approximation algorithm with a better constant performance ratio, unless \( P=NP \) [even for the unweighted case].

3. THE NEMHAUSER AND TROTTER LOCAL OPTIMIZATION ALGORITHM

Let \( G(V,E) \) be a simple graph. We denote by \( G(U) \) the subgraph of \( G \) induced by \( U \subseteq V \) and let \( U' = \{ u' \mid u \in U \} \). Define the weights of vertices in \( U' \) by \( w(u') = w(u) \).

Nemhauser and Trotter \([NT75]\) presented the following local optimization algorithm:

**Algorithm NT**

**Input:** \( G(V,E), w. \)

**Phase 1:** Define a bipartite graph \( B(V,V',E_B) \)
where \( E_B = \{(x,y') \mid (x,y) \in E \} \).

**Phase 2:** \( C_B \leftarrow C^*(B) \).

**Output:** \( C \leftarrow \{ x \mid x \in C_B \text{ AND } x' \in C_B \} \)
\( V_0 \leftarrow \{ x \mid x \in C_B \text{ XOR } x' \in C_B \} \)

The following theorem states results of Nemhauser and Trotter, but our proof does not use linear programming arguments.

\(^2\)Chvatal \([C79]\) gets, performance-ratio \( \leq \sum_{j=1}^{\Delta} \frac{1}{j} = O(\log \Delta) \).

[where \( \Delta \) is the maximum vertex degree in \( G \)].
The NT-Theorem: The sets $C_o, V_o$ which Algorithm NT produces, satisfies the following properties:

(i) If a set $D \subseteq V_o$ covers $G(V_o)$ then $C = D \cup C_o$ covers $G$.

(ii) There exists an optimum cover $C^*(G)$ such that $C^*(G) \supseteq C_o$.

[(i) and (ii) are called the local optimality conditions.]

(iii) $w(C^*(G(V_o))) \geq \frac{1}{2} w(V_o)$.

Proof: Define $I_o = \{ x \mid x \notin C_B \text{ AND } x \notin C_B \} = V - (V_o \cup C_o)$

Let $(x, y) \in E$. In order to prove (i) we need to show that either $x \in C$ or $y \in C$.

Case 1: $x \in I_o$. i.e. $x \notin C_B$. Thus, $y, y' \in C_o$ and therefore $y \in C_o$.

Case 2: $y \in I_o$. Same as Case 1.

Case 3: $x \in C_o$ or $y \in C_o$. This case is trivial.

Case 4: $x, y \in V_o$. Thus, either $x \in D$ or $y \in D$.

In order to prove (ii), let $S = C^*(G)$. Define $S_V = S \cap V_o, S_C = S \cap C_o, S_I = S \cap I_o$ and $S'_I = I_o - S_I$. Let us show that:

(*) $C_{B_1} = (V - S_I) \cup S_C'$ covers $B$.

Let $(x, y') \in E_B$. We need to show that either $x \in C_{B_1}$ or $y' \in C_{B_1}$.

Case 1: $x \notin S_I$. Thus, $x \in V - S_I \subseteq C_{B_1}$

Case 2: $x \in S_I$. Thus, $x \in I_o$ and therefore $x \notin S$ and $x \notin C_o$. It follows that $y \in S$ [since $S$ covers $(x, y)$] and $y \in C_o$. [by Case 1, in the proof of (i)]. Thus, $y \in S \cap C_o = S_C$ and therefore $y' \in S_C' \subseteq C_{B_1}$.

This proves (*). Now,

\[ w(V_o) + 2w(C_o') = w(V_o \cup C_o \cup C_o') \]

\[ = w(C_B) \quad \text{[by definitions of } V_o, C_o] \]

\[ \leq w(C_{B_1}) \quad \text{[by (*) and optimality of } C_B] \]

\[ = w((V - S_I) \cup S_C') \]

\[ = w(V_o \cup C_o \cup S_I \cup S_C'). \]
It follows that \( w(C_0) \leq w(S-S_Y) \). Thus, \( w(C_0 \cup S_Y) \leq w(S) \). However, \( C_0 \cup S_Y \) covers \( G \) [by (i)] and therefore \( C_0 \cup S_Y \) is an optimum cover of \( G \) and contains \( C_o \). This proves condition (ii).

In order to prove (iii), assume \( S_o \) is an optimum cover of \( G(V_o) \). By (i), \( C_0 \cup S_o \) covers \( G \), and by \( B \)'s definition \( C_o \cup C_o \cup S_o \cup S_o \) covers \( B \). Thus,

\[
\begin{align*}
  w(V_o) + 2w(C_o) &= w(C_B) \\
  &\leq w(C_o \cup C_o \cup S_o \cup S_o) \quad [\text{by optimality of } C_B] \\
  &= 2w(C_o) + 2w(S_o).
\end{align*}
\]

Therefore, \( w(V_o) \leq 2w(S_o) \).

Q.E.D.

Let us consider now the time-complexity of finding \( C^*(B) \), which determines the time-complexity of \( NT \).

For the unweighted case the problem can be converted into the maximum matching problem on \( B \) (see for example [BM76]) which is of time-complexity \( O(E\sqrt{V}) \) (see [HK73]).

For the weighted case the problem can be converted into a maximum flow problem (see, for example [HB1]) which is of time-complexity \( O(E^{2/3}V^{1/3}) \) [GI78] or \( O(EH\log V) \) [S80]. For a summary of the results, see Table 2.
4. PUTTING TOGETHER NT AND THE LOCAL-RATIO THEOREM

Hochbaum [H81] suggested the following approach to approximate WVC: Let 
\( G(V,E),w \) be the problem's input, such that \( w(C^*(G)) \geq \frac{k}{\Delta} w(V) \). (This is achieved by the \( NT \) algorithm.) Color \( G \) by \( k \) colors and let \( I \) be the "heaviest" monochromatic set of vertices. The cover produced is \( C = V - I \). It follows that

\[
\frac{w(C)}{w(C^*)} = \frac{w(V) - w(I)}{w(C^*)} \leq \frac{w(V) - w(V)/k}{\frac{k}{\Delta} w(V)} = 2 - \frac{2}{\Delta}.
\]

For general graphs she gets the ratio \( 2 - \frac{2}{\Delta} \) (\( \Delta \) is the maximum degree) and for planar graphs (\( k = 4 \)) the performance ratio \( \leq 1.5 \) in time-complexity of \( NT \) and 4-coloring.

We suggest the use of a preparatory algorithm in which all triangles are omitted (with local-ratio 1.5) and therefore, the residual graph is easier to color.

**Algorithm COVER2**

**Input:** \( G(V,E),w,k \).

**Phase 1:** [Triangle elimination]

For every triangle \( T \ (T \subseteq V) \) do

Find \( \delta = \min \{ w(x) | x \in T \} \)

For every \( x \in T \) do \( w(x) \leftarrow w(x) - \delta \) end

\( C_1 = \{ x | w(x) = 0 \} \)

\( V_1 = V - C_1 \).

**Phase 2:** Call \( NT(G(V_1),w) \) to get \( C_2, V_0 \).

**Phase 3:** Find a cover approximation, \( C_3 \), of \( G(V_0) \) by using \( k \)-coloring (as in Hochbaum's approach).

**Output:** \( C = C_1 \cup C_2 \cup C_3 \).
contains of a single graph \( G \) which is a triangle), the local-ratio of Phase 2 is 1 and the local-ratio of Phase 3 is \( 2 - \frac{2}{k} \).

Since the graph colored in Phase 3 is triangle-free, we get the following additional results:

1. For general graphs \( r_{\text{COVER}}(n) \leq 2 - \frac{1}{\sqrt{n}} \) by using Wigderson's approach \([W82a]\) for coloring a triangle-free graph by \( k = 2\sqrt{n} \) colors in linear time \([W82b]\).
2. For planar graphs \( r_{\text{COVER}}(n) \leq 1.5 \), and the time-complexity of 4-Coloring a triangle-free planar graph is linear. (One uses the fact that in such graphs there is always a vertex of degree \( \leq 3 \); See, for example, \([Ha72]\).) This prevents the need to use a more complex 4-Coloring algorithms.

Before we present our main algorithm we need a few preliminaries.

Let the triple \( (G(V,E),w,k) \) (where \( G \) is a graph with weight function \( w \) and \( k \) is a positive integer) be called Proper if the following conditions hold:

1. \( (2k-1)^k \geq |V| \).
2. There are no odd circuits of length \( \leq 2k - 1 \).
3. \( w(C^*) \geq \frac{1}{k} w(V) \).

In the following procedure the statements in square brackets are added for the analysis only, and variables \( j \) (integer), \( C_0, V_0, C_1, V_1, \ldots \) (sets), are used only in the brackets.
Procedure $COVER.PROPER$

Input: \(\text{proper (} G(V,E), w, k \text{)}\).

Phase 0: \(V' \rightarrow V, C \leftarrow \emptyset, [j \leftarrow 0]\)

Phase 1: While \(V' \neq \emptyset\) do

\[
\text{Find } v \in V', s.t. \ w(v) = \max \{ w(u) | u \in V' \}. \\
\text{Let } A_0, A_1, \ldots, A_k \text{ be the first } k+1 \text{ layers}\uparrow \\
\text{of a Breadth-First-Search (BFS) on } G(V') \text{ starting with } A_0 = \{v\}. \\
\text{Define } B_{2t} = \bigcup_{i=0}^{t} A_{2i} \text{ and } B_{2t+1} = \bigcup_{i=0}^{t} A_{2i+1} \text{ (for } t = 0, 1, 2, \ldots). \\
f \leftarrow \min \{ s \mid w(B_s) \leq (2k-1)w(B_{s-1}) \}. \\
\text{Add } B_f \text{ to } C'. \ [C_f \leftarrow B_f] \\
\text{Remove } B_f \cup B_{f-1} \text{ from } V'. \ [V_f \leftarrow B_f \cup B_{f-1}, \ j \leftarrow j + 1]
\]

end

Output: \(C \leftarrow C'.\)

Proposition 1: Procedure $COVER.PROPER$ satisfies the following properties:

1. In every application of Phase 1, \(f \leq k\).
2. In every application of Phase 1, \(B_{f-1}\) is an independent set in \(G(V')\).
3. \(C\) covers \(G\).
4. For every iteration \(j\), \(w(C_j) \leq (1 - \frac{1}{2k}) w(V_j)\).
5. \(w(C) \leq (1 - \frac{1}{2k}) w(V)\).
6. \(R_{COVER.PROPER}(G) \leq 2 - \frac{1}{k}\).
7. The time complexity is \(O(|V| \log |V| + |E|)\).

Proof: 

\("\uparrow\)Starting with some \(m\), \(A_m, A_{m+1}, \ldots, A_k\) may be empty.
(1) Assume the contrary. Thus, for every \( s \leq k \), \( w(B_s) > (2k-1) \cdot w(B_{s-1}) \). Thus,

\[
w(B_k) > (2k-1)^k \cdot w(B_0) \quad \text{[by the assumption]}
\]

\[
\geq |V| \cdot w(B_0) \quad \text{[by (1) of the definition of proper]} \\
= |V| \cdot w(v) \quad [B_0 = A_0 = \{v\}] \\
\geq |V| \cdot w(v) \quad [V \subseteq V] \\
\geq w(V) \quad \text{[by } v \text{'s definition]} \\
\geq w(B_k) \quad [B_k \subseteq V]
\]

which is absurd.

(2) An edge between two vertices of \( A_k, s < k \), implies the existence of an odd-circuit of length \( \leq 2k-1 \). This contradicts condition (ii) of properness.

(3) For every iteration \( j \), all edges which are (indirectly) deleted are covered by the current \( C_j \), which joins \( C \). This follows from the properties of BFS and (2) above.

(4) For every iteration \( j \), \( w(B_j) \leq (2k-1) \cdot w(B_{j-1}) \). Since \( B_j = C_j \), and \( B_{j-1} = V_j - C_j \), we have \( w(C_j) \leq (2k-1) \cdot [w(V_j) - w(C_j)] \). This implies the stated inequality.

(5) By summation of the inequality of (4) for every \( j \).

(6) By definition \( R_{\text{COVER.PROPER}}(G,w) = \frac{w(C)}{w(C')} \). Property (5) above and condition (iii) of properness imply that

\[
R_{\text{COVER.PROPER}}(G,w) \leq \frac{(1 - \frac{1}{2k}) \cdot w(V)}{\frac{1}{2} \cdot w(V)}.
\]

(7) We may start the algorithm by sorting the vertices according to their weights, which requires \( O(|V| \log |V|) \) steps. This complexity includes, now, the total time used in Phase 1 for finding \( \text{Max} \{w(u) \mid u \in V\} \).

It is not necessary to continue the BFS of Phase 1, beyond layer \( f \). Thus, each such search is linear in the num of edges to be eliminated from \( G(V) \), and the total time spend in the search of Phase 1 is linear in \( |E| + |V| \).

Thus (7) follows.

Q.E.D.
Now, the main algorithm.

Algorithm COVER3

**Input**: \( G(V,E), w \).

**Phase 0**: Find the least integer \( k \) s.t. \((2k-1)^k \geq |V|\).

**Phase 1**: Elimination of short odd circuits with local-ratio <= \( 2-\frac{1}{k} \).

For every odd circuit \( D \subset V \) s.t. \( |D| \leq 2k-1 \) do

Let \( \delta = \min \{ w(x) | x \in D \} \)

For every \( x \in D \) do \( w(x) \leftarrow w(x) - \delta \) end

**end**

\( C_1 \leftarrow \{ x | w(x) = 0 \} \).

\( V_1 \leftarrow V - C_1 \).

**Phase 2**: Call \( NT(G(V_1), w) \) to get \( C_0, V_0 \).

**Phase 3**: Call \( COVER.PROPER(G(V_0), w, k) \) to get \( C_2 \).

**Output**: \( C = C_1 \cup C_0 \cup C_2 \).

**Proposition 2**: Algorithm \( COVER3 \) satisfies the following properties:

1. \( \tau_{COVER}(n) \leq 2 - \frac{1}{k} \).
2. Its time complexity is the same as \( NT \)'s. (see Table 2)

(3) For unweighted graphs, its time complexity is \( O(|V| \cdot |E|) \).

**Proof**:

1. The combination of Phase 2 and 3 yields an algorithm with performance ratio \( \leq 2 - \frac{1}{k} \), since the \( NT \) algorithm performs local-optimization (ratio=1) and, for proper graphs, \( COVER.PROPER \) has performance ratio \( \leq 2 - \frac{1}{k} \) by [Proposition 1(3)]. Consider the Local-Ratio-Theorem, let \( G_i \) be a simple circuit length \( 2l-1 \) thus, \( n_i = 2l-1 \) and \( e_i = l \), therefore, \( \bar{n}_i = (2l-1)/l = 2-1/l \). Now let us consider the Generalized-Local-Ratio-Theorem were, \( G = \{ G_i | l \leq k \} \), thus,
\( r_T = \text{Max} \{ r_i : 1 \leq k \} = 2^{-1/k} \) and (1) follows.

(2) Let us perform Phase 1 in a slightly different way: Choose a vertex \( v \), and build the first \( k \) layers of the BFS starting from \( v \). If there is an edge \( u-w \), where \( u \) and \( w \) belong to the same layer, then an odd-length-circuit \( D \) has been detected (see for example [IR78]). In this case we find \( \delta = \text{Min} \{ w(x) | x \in D \} \), reduce the weights of the vertices in \( D \) by \( \delta \) and the vertices whose weight is zero are eliminated from the representation of the graph (for the purpose of performing Phase 1). If no such edge (closing an odd circuit) is detected, then \( v \) is eliminated from the representation of the graph, since no odd-circuit of length \( \leq 2k-1 \) passes through it. In any case, at least one vertex is eliminated for each BFS. Thus the time complexity is \( O(|V| \cdot |E|) \).

In Phase 2, the best known time-complexity of the NT algorithm (table 2) is greater than \( |V| \cdot |E| \). Phase 3 requires \( O(|E| + |V| \log |V|) \), by Proposition 1(7).

Thus, the whole algorithm is of time complexity as the NT algorithm.

(3) For unweighted graphs, the NT algorithm can be performed in \( O(|E| \sqrt{|V|}) \) time and therefore, the algorithm is of time complexity \( O(|V| \cdot |E|) \).

\[ r_T = \text{Max} \{ r_i : 1 \leq k \} = 2^{-1/k} \] and (1) follows.

(2) Let us perform Phase 1 in a slightly different way: Choose a vertex \( v \), and build the first \( k \) layers of the BFS starting from \( v \). If there is an edge \( u-w \), where \( u \) and \( w \) belong to the same layer, then an odd-length-circuit \( D \) has been detected (see for example [IR78]). In this case we find \( \delta = \text{Min} \{ w(x) | x \in D \} \), reduce the weights of the vertices in \( D \) by \( \delta \) and the vertices whose weight is zero are eliminated from the representation of the graph (for the purpose of performing Phase 1). If no such edge (closing an odd circuit) is detected, then \( v \) is eliminated from the representation of the graph, since no odd-circuit of length \( \leq 2k-1 \) passes through it. In any case, at least one vertex is eliminated for each BFS. Thus the time complexity is \( O(|V| \cdot |E|) \).

In Phase 2, the best known time-complexity of the NT algorithm (table 2) is greater than \( |V| \cdot |E| \). Phase 3 requires \( O(|E| + |V| \log |V|) \), by Proposition 1(7).

Thus, the whole algorithm is of time complexity as the NT algorithm.

(3) For unweighted graphs, the NT algorithm can be performed in \( O(|E| \sqrt{|V|}) \) time and therefore, the algorithm is of time complexity \( O(|V| \cdot |E|) \).

Q.E.D.

**Corollary:** \( r_{\text{COVERS}}(n) < 2^{-\frac{\log \log n}{2 \log n}} \).

**Proof:** Define \( g(n) = \frac{\log n}{\log \log n} \) which is monotone increasing for \( n \geq 16 \). By \( k \)-th definition \((2k-3)^k<n\). Thus, \( g((2k-3)^k-1) < g(n) \).

We want to show that \( k < 2g(n) \). Thus, it suffices to show that \( k < 2g((2k-3)^k-1) \).

This is an exercise in elementary mathematics.

Q.E.D.
REFERENCES


