Original Notes by Chi-Jiun Su.

6 Vertex Cover

The vertex cover problem is defined as follows:

> *Given a graph G= (V,E) find a subset $C \subseteq V$ such that for all $(u, v) \in E$, at least one of u or v is included in C and the cardinality of set C is minimized.*

This problem is a special case of the set cover problem in which the elements correspond to edges. There is a set corresponding to each vertex which consists of all the edges incident upon it and each element is contained exactly in two sets. Although the greedy algorithm for set cover we mentioned in the last class can be applied for vertex cover, the approximation factor is still $H(\Delta(G))$, where $\Delta(G)$ is the maximum degree of a node in the graph G. Can we hope to find algorithms with better approximation factors? The answer is yes and we will present a number of 2-approximation algorithms.

6.1 2-Approximation Algorithms

6.1.1 Simple Algorithm

Input: Unweighted graph G(V,E)
Output: Vertex cover C

1. $C \leftarrow \emptyset$
2. **while** $E \neq \emptyset$ **do begin**
   Pick any edge $e = (u, v) \in E$ and choose *both* end-points u and v.
   $C \leftarrow C \cup \{u, v\}$
   $E \leftarrow E \setminus \{e \in E \mid e = (x, y) \text{ such that } u \in \{x, y\} \text{ or } v \in \{x, y\}\}$
   end.
3. **return** C.

The algorithm achieves an approximation factor of 2 because for a chosen edge in each iteration of the algorithm, it chooses both vertices for vertex cover while the optimal one will choose at least one of them.

Note that picking only one of the vertices arbitrarily for each chosen edge in the algorithm will fail to make it a 2-approximation algorithm. As an example, consider a star graph. It is possible that we are going to choose more than one non-central vertex before we pick the central vertex.

Another way of picking the edges is to consider a maximal matching in the graph.

6.1.2 Maximal Matching-based Algorithm

Input: Unweighted graph G(V,E)
Output: Vertex cover C

1. Pick any maximal matching $M \subset E$ in G.
2. $C \leftarrow \{v \mid v \text{ is matched in } M\}$.
3. **return** C.

It is also a 2-approximation algorithm. The reason is as follows. Let M be a maximal matching of the graph. At least one of the end-points in all the edges in $E \setminus M$ is matched since, otherwise, that edge could be added to M, contradicting the fact that M is a maximal matching. This implies that every edge in E has at least one end-point that is matched. Therefore, C is a vertex cover with exactly $2\ |M|$ vertices.

To cover the edges in M, we need at least $|M|$ vertices since no two of them share a vertex. Hence, the optimal vertex cover has size at least $|M|$ and we have

$$VC = 2\ |M| \ \leq \ 2\ VC^{optimal}$$

where VC is the size of vertex cover.

7 Weighted Vertex Cover

Given a graph $G = (V, E)$ and a positive weight function $w : V \to R^+$ on the vertices, find a subset $C \subseteq V$ such that for all $(u, v) \in E$, at least one of u or v is contained in C and $\sum_{v \in C} w(v)$ is minimized.

The greedy algorithm, which choose the vertex with minimum weight first, does not give a 2-approximation factor. As an example, consider a star graph with $N + 1$ nodes where the central node has weight $1 + \epsilon$, $\epsilon << 1$, and other non-central nodes have weight 1. Then, it is obvious that the greedy algorithm gives vertex cover of weight N while the optimal has only weight $1 + \epsilon$.

7.1 LP Relaxation for Minimum Weighted Vertex Cover

Weighted vertex cover problem can be expressed as an integer program as follows.

$$\begin{array}{rccc} \text{OPT-VC} \ = & \text{Minimize} & \sum_{v \in V} w(v)X(v) & \\ \text{subject to} & & X(u) + X(v) \geq 1 & \forall e = (u, v) \in E \\ \text{where} & & X(u) \in \{0, 1\} & \forall u \in V \end{array}$$

If the solution to the above problem has $X(u) = 1$ for a vertex u, we pick vertex u to be included in the vertex cover.

But, since the above integer program is NP-complete, we have to relax the binary value restriction of $X(u)$ and transform it into a linear program as follows.

$$\begin{array}{rccc} \text{LP-Cost} \ = & \text{Minimize} & \sum_{v \in V} w(v)X(v) & \\ \text{subject to} & & X(u) + X(v) \geq 1 & \forall e = (u, v) \in E \\ \text{where} & & X(u) \geq 0 & \forall u \in V \end{array}$$

Nemhauser and Trotter showed that there exists an optimal solution of the linear program of the form: $X^*(u) = \begin{cases} 0 \\ \frac{1}{2} \\ 1 \end{cases}$.

How do we interpret the results from LP as our vertex cover solution? One simple way is to include a vertex u in our vertex cover if $X^*(u) \geq \frac{1}{2}$. That is, the resultant vertex cover is

$$C = \{u \in V | X^*(u) \geq \frac{1}{2}\}$$

This forms a cover since for each edge, it cannot be that the X values of both end points is $< \frac{1}{2}$. This turns out to give 2-approximation of minimum weighted vertex cover. Why?

$$\sum_{u \in C} w(u) \ \leq \ \sum_{u \in C} 2X^*(u)w(u) \ \leq \ \sum_{u \in V} 2X^*(u)w(u) \ = \ 2 \text{ LP-cost}$$

Moreover, since LP is obtained by the relaxation of binary value restriction of $X(u)$ in the original integer program,

$$\text{LP-Cost} \le \text{OPT-VC}.$$

Therefore, the cost of the vertex cover we rounded from the LP is

$$\text{VC} = \sum_{u \in C} w(u) \;\le\; 2 \text{ OPT-VC}$$

7.1.1 Primal-Dual Applied to Weighted Minimum Cover

Although relaxation to LP and rounding from the LP solutions gives us a 2-approximation for the minimum weighted vertex cover, LP is still computationally expensive for practical purposes. How about the dual form of the relaxed LP?

Given a LP in standard form:

Primal (**P**)

$$\begin{aligned} \text{Minimize} \quad & c^T x \\ \text{subject to} \quad & Ax \ge b \\ & x \ge 0 \end{aligned}$$

where x, b, c are column vectors, A is a matrix and T denotes the transpose. We can transform it into its dual form as follows:

Dual (**D**)

$$\begin{aligned} \text{Maximize} \quad & b^T y \\ \text{subject to} \quad & A^T y \le c \\ & y \ge 0 \end{aligned}$$

Then, the dual of our relaxed LP is

$$\begin{aligned} \text{DLP-Cost} \;=\; & \quad \text{Maximize} \quad \textstyle\sum_{e \in E} Y(e) \\ \text{subject to} \quad & \textstyle\sum_{u:e=(u,v)\in E} Y(e) \;\le\; w(v) \quad \forall v \in V \\ & Y(e) \;\ge\; 0 \qquad\qquad \forall e \in E \end{aligned}$$

Y_e is called "*packing function*". A packing function is any function $Y : E \to R^+$ which satisfies the above two constraints.

According to *Weak Duality Theorem*, if PLP-Cost and DLP-Cost are the costs for a *feasible* solution to the primal and dual problem of the LP respectively,

$$\text{DLP-Cost} \;\le\; \text{PLP-Cost}$$

Therefore,

$$\text{DLP-Cost} \;\le\; \text{DLP-Cost}^* \;\le\; \text{PLP-Cost}^*$$

But, remember that PLP-Cost* $\le$ OPT-VC. Therefore, we don't need to find the optimal solution to the dual problem. Any feasible solution is just fine, as long as we can upper bound the cost of the vertex cover as a function of the dual feasible solution. How do we find a feasible solution to the dual problem? It reduces to the problem: "how to find a packing function". In fact, any *maximal packing function* will work.

By *maximal*, we mean that for an edge $e(u, v) \in E$, increase $Y(e)$ as much as possible until the inequality in the first constraint becomes equality for u or v. Suppose u is an end-point of the edge e for which the constraint becomes equality. Add u to the vertex cover, C, and remove all the edges incident on u. Obviously, C is a vertex cover. Does this vertex cover give 2-approximation?

Note that

$$w(v) \;=\; \sum_{e(u,v):u\in N(v)} Y(e) \qquad \forall v \in C$$

where $N(v)$ is the set of vertices adjacent to vertex v. Therefore,

$$\sum_{v \in C} w(v) \;=\; \sum_{v \in C} \sum_{e(u,v): u \in N(v)} Y(e) \;=\; \sum_{e=(u,v) \in E} |C \cap \{u,v\}| \, Y(e) \;\leq\; \sum_{e(u,v) \in E} 2\, Y(e)$$

However, according to the constraint of the packing function,

$$w(v) \;\geq\; \sum_{e(u,v): u \in N(v)} Y_e$$

$$\sum_{v \in C^*} w(v) \;\geq\; \sum_{v \in C^*} \sum_{e(u,v): u \in N(v)} Y_e$$

where C^* is the optimal weighted vertex cover.

Since some of the edges are counted twice in the right-side expression of the above inequality,

$$\sum_{v \in C^*} \sum_{e(u,v): u \in N(v)} Y_e \;\geq\; \sum_{e \in E} Y_e$$

Hence,

$$\sum_{v \in C} w(v) \;\leq\; 2 \sum_{v \in C^*} w(v) \;=\; 2 \text{ OPT-VC}$$

One simple packing function is, for some vertex v, to distribute its weight $w(v)$, onto all the incident edges *uniformly*. That is, $Y_e = \frac{w(v)}{d(v)}$ for all the incident edges e where $d(v)$ is the degree of the vertex v. For other vertices, adjust Y_e appropriately. This way of defining packing function leads to the following algorithm.

7.1.2 Clarkson's Greedy Algorithm

Input: Graph G(V,E) and weight function w on V
Output: Vertex cover C

1. **for all** $v \in V$ **do** $W(v) \leftarrow w(v)$
2. **for all** $v \in V$ **do** $D(v) \leftarrow d(v)$
3. **for all** $e \in E$ **do** $Y_e \leftarrow 0$
4. $C \leftarrow \emptyset$
5. **while** $E \neq \emptyset$ **do begin**
   Pick a vertex $v \in V$ for which $\frac{W(v)}{D(v)}$ is minimized
   for all edges $e = (u,v) \in E$ **do begin**
   $E \leftarrow E \setminus e$;
   $W(u) \leftarrow W(u) - \frac{W(v)}{D(v)}$ and $D(u) \leftarrow D(u) - 1$
   $Y_e \leftarrow \frac{W(v)}{D(v)}$
   end
   $C \leftarrow C \cup \{v\}$ and $V \leftarrow V \setminus \{v\}$
   $W(v) \leftarrow 0$
   end.
6. **return** C.

7.1.3 Bar-Yehuda and Even's Greedy Algorithm

Input: Graph G(V,E) and weight function w on V
Output: Vertex cover C

1. **for all** $v \in V$ **do** $W(v) \leftarrow w(v)$
2. **for all** $e \in E$ **do** $Y_e \leftarrow 0$
3. $C \leftarrow \emptyset$
4. **while** $E \neq \emptyset$ **do begin**
   Pick an edge $(p, q) \in E$
   Suppose $W(p) \leq W(q)$
   $Y_e \leftarrow W(p)$
   $W(q) \leftarrow W(q) - W(p)$
   for all edges $e = (p, v) \in E$ **do begin**
   $E \leftarrow E \setminus e$;
   end
   $C \leftarrow C \cup \{p\}$ and $V \leftarrow V \setminus \{p\}$
   end.
5. **return** C.

Note that both these algorithms come up with a maximal packing (or maximal dual solution) and pick those vertices in the cover for which the dual constraint is met with equality. Any method that comes up with such a maximal dual solution would in fact yield a 2 approximation. These are just two ways of defining a maximal dual solution.