1 Minimum Feedback Vertex Set Problem

The minimum feedback vertex set problem is an $\mathcal{NP}$-complete graph problem that is defined as follows. Given a graph $G = (V, E)$ and a weight $w(v_i)$ for every vertex v_i, find a subset $S \subseteq V$, such that the induced subgraph $G[V - S]$ is acyclic and $\sum_{v_i \in S} w(v_i)$ is minimized. The proof and the arguments described in this work follow closely Chapter 6 in Vazirani's book *Approximation Algorithms*.

2 Cyclomatic Number of a graph

Let $G = (V, E)$ be a connected graph.

2.1 Notation and Definitions

Simple cycle. A cycle where no vertex repeats.

Characteristic vector of a cycle c. An element of $GF[2]^{|E|}$ where the ith element is set to 1 iff $e_i \in c$. The characteristic vectors of two distinct simple cycles are linearly independent, because the cycles must differ in at least one edege.

Cycle space $S(G)$ of a graph G. Let $C(G)$ be the set of all characteristic vectors for all simple cycles in G. We define $S(G) = \text{span}(C(G))$. Note that, $S(G) \leq GF[2]^{|E|}$.

Cyclomatic number. $cyc(G) = \dim S(G)$.

Connected components. $\kappa(G)$ is the number of connected components in G.

Decrease in cyclomatic number. $\delta_G(v) = cyc(G) - cyc(G[V - \{v\}])$ is the decrease of the cyclomatic number of G when removing v from G.

Cyclomatic weight function. $w(v)$ is a cyclomatic weight function if $w(v) = c \cdot \delta_G(v)$ for some $c > 0$.

2.2 Computation of the cyclomatic number

We claim that the following formulas holds true for every graph G.

Lemma 1 $cyc(G) = |E| - |V| + \kappa(G)$.

Proof For the moment, let G be a graph consisting of a single connecting component. We show that $cyc(G) = |E| - |V| + 1$.

The dimension of a vector space is the number of base vectors, i.e. the number of linearly independent vectors. Therefore, $cyc(G)$ is the number of simple cycles in G. Let T be a spanning tree of G. For every $e \in E$ with $e \notin T$, $T \cup \{e\}$ has exactly one simple cycle, and all these simple cycles are distinct, since they all have a different edge e. Therefore, the characteristic vectors for all these simple cycles are linearly independent. T has $|V| - 1$ edges, therefore, there are $|E| - |V| + 1$ edges e that we could add to T, and $cyc(G) \geq |E| - |V| + 1$.

For every spanning tree T, we can generate a cut of the graph, i.e. a partitioning of V into V_1 and V_2, by removing an edge $e \in T$. There are $|V| - 1$ such cuts, because the spanning tree has $|V| - 1$ edges. Similarly to the characteristic vector of a cycle, we can define the characteristic vector of cut as follows: it is an element of $GF[2]^{|E|}$, where an element is 1, if the corresponding edge is part of the cut. Note, that every cycle in G must cross the cut an even number of times. Otherwise, the *cycle* would get stuck on one side of the cut and not return to the beginning of the cycle. Therefore, the space of characteristic vectors of cuts is orthogonal to the cycle space and the following argument holds true.

- There are $|V| - 1$ cuts, resulting in $|V| - 1$ characteristic vectors for T.
- All characteristic vectors for T are linearly independent, since every distinct cut lacks a different edge.
- $cyc(G) \leq |E| - |V| + 1$, since both vector spaces are orthogonal.
- $cyc(G) = |E| - |V| + 1$, since also $cyc(G) \geq |E| - |V| + 1$.

The cycle space of a graph consisting of multiple connected components is the sum of the cycle spaces of every connected component. For every connecting component, we can apply the proof shown above, resulting in a total cycle space of $\sum_{c \in components(G)} |E[c]| - |V[c]| + 1 = |E| - |V| + |components(G)| = |E| - |V| + \kappa(G)$.

Lemma 2 *$cyc(G) = \sum_{i=1}^{|F|} \delta_{G_{i-1}}(v_i)$, where $G_i = G[V - \{v_1, v_2, \ldots, v_i\}]$, i.e. the vertex-induced subgraph when removing the first i vertices of the feedback set F.*

Proof The proof follows from the definition of F. $G[V - F]$ is acyclic, i.e. when we remove all vertices F, we must have destroyed all simple cycles in G, and there are exactly $cyc(G)$ simple cycles in G.

Lemma 3 *If $H \leq G$, i.e. H is a subgraph of G, then $\forall v \in V : \delta_H(v) \leq \delta_G(v)$.*

Proof Let us, without loss of generality, assume that G and H are connected. We know that $\delta_G(v) = \deg_G(v) - \kappa(G[V - \{v\}])$, therefore we can proove the lemma by showing that $\deg_H(v) - \kappa(H[V - \{v\}]) \leq \deg_G(v) - \kappa(G[V - \{v\}])$. Now, consider an edge $e \in E[G - H]$.

- First case: e is connected to v. Removing v might generate an additional component, i.e. increase κ by one. However, it also increases $\deg_G(v)$ by one.
- Second case: e is not connected to v. In that case, e does not affect the degree of v, but it might reduce the number of components in case it connects two otherwise disconnected components.

Lemma 4 *$cyc(G) \leq \sum_{v \in F} \delta_G(v)$*

Proof We know that $cyc(G) = \sum_{i=1}^{|F|} \delta_{G_{i-1}}(v_i)$. Note, that G_i is a subgraph of G_{i-1}. Due to the previous lemma, we know that $\delta_{G_i}(v) \leq \delta_{G_{i-1}}(v)$. Therefore, $\sum_{i=1}^{|F|} \delta_{G_{i-1}}(v_i) = cyc(G) \leq \sum_{i=1}^{|F|} \delta_{G_0}(v_i) = \sum_{v \in F} \delta_G(v)$.

Lemma 5 *If $w(v)$ is a cyclomatic weight function and F is an optimal feedback vertex set, then $c \cdot cyc(G) \leq OPT$.*

Proof We know that $cyc(G) \leq \sum_{v \in F} \delta_G(v)$, therefore $c \cdot cyc(G) \leq c \cdot \sum_{v \in F} \delta_G(v) = \sum_{v \in F} c \cdot \delta_G(v) = w(F) = OPT$, where c is the constant of the cyclomatic weight function.

Lemma 6 *$\sum_{v \in F} \delta_G(v) \leq 2 \cdot cyc(G)$ if F is a minimal feedback vertex set. Also, $w(F) \leq 2 \cdot OPT$*

Proof The proof for this lemma can be found in the next lecture notes or in the book *Approximation Algorithms* by Vazirani.

3 Layering Algorithm

We now consider graphs with an arbitrary weight function. For a graph G, we define c as follows.

$$c = \min_{v \in V} \frac{w(v)}{\delta_G(v)}$$

We now split $w(v)$ into two parts. $w(v) = w'(v) + t(v)$, where $t(v) = c \cdot \delta_G(v)$. Note, that $w'(v) = 0$ for at least one vertex v. In the following, we present an iterative algorithm that generates a sequence of graphs, until, a some point, the graph is acyclic. We generate a subsequent graph by removing vertices from the previous one, therefore $G_i \leq G_{i+1}$ and $G = G_0$.

The algorithm for generating a minimal feedback vertex set is based on the following idea. If $G = (V, E)$, $H = (V', E')$, $H \leq G$, and F is a minimal feedback vertex for H, then we can generate a minimal feedback vertex for G by taking all vertices from F and adding some additional vertices from $V - V'$.

Lemma 7 *Let $G = (V, E)$, $H = (V', E')$, $H \leq G$, and F be a minimal feedback vertex set for H. Then, $F \cup F'$ is a minimal vertex set for G, where $F' \subseteq V - V'$, such that $F \cup F'$ is a feedback vertex set for G and F' is a minimal set.*

Proof Let $v \in F$ an arbitrary vertex. There must be a cycle C in H that uses v but no other vertex from F. Otherwise, we could remove one of these two vertices and F would be minimal. Since $F' \subseteq V - V'$, we know that $F' \cap V' = \emptyset$. Therefore, C uses only the vertex v from $F \cup F'$. Therefore, if we would remove v from $F \cup F'$, this set would no longer be a feedback vertex set. Therefore, this set is minimal.

Algorithm We iteratively build a sequence of graphs as follows.

- $G_0 \leftarrow G$, $V_0 \leftarrow V$, $w'_0 \leftarrow w$, $i \leftarrow 0$
- While G_i is not acyclic
  - $c \leftarrow \min_{v \in V_i} \frac{w'_i(v)}{\delta_{G_i}(v)}$
  - $\forall v \in V_i : t_i(v) \leftarrow c \cdot \delta_{G_i}(v)$
  - $\forall v \in V_i : w'_{i+1}(v) \leftarrow w'_i(v) - t_i(v)$
  - $V_{i+1} = \{v \in V_i \mid w'_{i+1} > 0\}$
  - $G_{i+1} \leftarrow G_i[V_{i+1}]$
  - $i \leftarrow i + 1$
- $k \leftarrow i$, $H \leftarrow G_k$

Based on the graph G_i, we can build a feedback vertex set as follows. Start with $F_k = \emptyset$. For every subsequent graph $G_{k-1}, \ldots, G_0 = G$, add a minimal set of vertices from $V_{i-1} - V_i$ (i.e., the vertices that we add when considering the next graph) to the feedback vertex set, such that the new set is a feedback vertex set for the next graph.

Lemma 8 *The algorithm described above approximates the optimal solution by a factor of* 2.

Let us assume that F^* is an optimal solution for the graph G. Consider any vertex-induced subgraph G_i. $F^* \cap V_i$ is a feedback vertex set for G_i. Note, that for every vertex v, $\sum_{i=0}^{k} t_i(v) = w(v)$, because $t_i(v) = w'_i(v) - w'_{i+1}(v)$ in the algorithm. Therefore, the following term holds true, where OPT_i is the weight of an optimal vertex feedback set for G_i.

$$OPT = w(F^*) = \sum_{i=0}^{k} t_i(F^* \cap V_i) \geq \sum_{i=0}^{k} OPT_i$$

We can use the same technique for F_0. $F_0 \cap V_i$ is an optimal vertex feedback set for G_i.

$$w(F_0) = \sum_{i=0}^{k} t_i(F_0 \cap V_i) = \sum_{i=0}^{k} t_i(F_i)$$

With Lemma 5 and Lemma 6 we can conclude that $w(F_0) \leq 2 \sum_{i=0}^{k} OPT_i \leq 2OPT$.