CMPUT 675: Topics in Combinatorics and Optimization **Fall 2016**

Lecture 6 (Sept. 19): Gomory-Hu Trees

Lecturer: Zachary Friggstad *Scribe: Mirmahdi Rahgoshay*

6.1 Algorithm for Constructing Gomory-Hu Trees

In this lecture we continued the discussion on Gomory-Hu trees, which are a compact way to represent $s-t$ cuts for all pairs s, t. We provide an algorithm to construct the tree. First we recall two lemmas from previous lecture.

Lemma 1 *For any sequence of $k \geq 2$ distinct vertices $v_1, v_2, ..., v_k$ we have:*

$$\lambda_{v_1,v_k} \geq \min_{1 \leq i \leq k} \lambda_{v_i,v_{i+1}}. \tag{6.1}$$

(See previous lecture for complete proof)

Lemma 2 *Let s, $t \in V$ be distinct vertices and A a minimum $s-t$ cut. For every distinct vertices u, $v \notin A$ there is a $u-v$ minimum cut B with $A \subseteq B$ or $A \cap B = \emptyset$.*

(See previous lecture for complete proof)

In the following we are going to provide an algorithm to construct a Gomory-Hu Tree for an undirected graph after recalling the definition of the Gomory-Hu tree.

Definition 1 *A* ***Gomory-Hu Tree (GHT)*** *for an undirected graph $\mathcal{G} = (V; E)$ with capacities μ is a tree $\mathcal{T} = (V; F)$ on the same set of vertices as $\mathcal{G}$ such that for every $uv \in F$, the fundamental cut C_{uv} is a minimum $u-v$ cut in $\mathcal{G}$ (i.e. $\mu(\delta(C_{uv})) = \lambda_{u,v}$).*

Basic idea: Select vertex pair $s, t \in V(G)$ and find a minimum s-t-cut A. Contract A (or $B := V - A$) to one vertex. Then select s', $t' \in B$ (or A). Find a minimal s'-t'-cut A' in contracting graph G' with $V(G') - A'$. Observe that, based on the lemma 2, a minimal cut in the contracting graph corresponds to a minimal cut in the original graph. Recurse until we reach a graph where there is only one vertex in the uncontracted part.

Now we describe the algorithm which constructs a Gomory-Hu tree. At each step, the vertices of the tree $\mathcal{T}$ being constructed are subsets of V and, furthermore, constitute a partition of V. Throughout we let $V(\mathcal{T})$ and $E(\mathcal{T})$ denote the vertices of the tree $\mathcal{T}$ being constructed.

A key operation is that of **contracting**. It's probably easier to see what this is by simply looking at Figure 6.1, but the formal definition follows. Given disjoint subsets $S_1, S_2, \ldots, S_k$ of V, let $\mathcal{G}/S_1, S_2, \ldots, S_k$ be the graph with vertices $(V - \cup_{i=1}^{k} S_i) \cup \{v_{S_i} : 1 \leq i \leq k\}$ where the vertices v_{S_i} are new vertices representing the subsets S_i. For each vertex $v \in V$, let $\phi(v)$ be v if $v \notin \cup_{i=1}^{k} S_i$, otherwise let $\phi(v) = v_{S_i}$ where i is such that $v \in S_i$. The edges of the contracted graph are $\{\phi(u)\phi(v) : uv \in E, \phi(u) \neq \phi(v)\}$ and each such edge $\phi(u)\phi(v)$ has capacity $\mu(uv)$.

Algorithm 1 Gomory-Hu Algorithm

Input: Undirected graph $\mathcal{G} = (V; E)$, $\mu : E \to \mathbb{R}_{\geq 0}$.
Output: A Gomory-Hu tree $\mathcal{T}$ for $(\mathcal{G}, \mu)$.

```
V(T) ← {V(G)} (a single vertex that corresponds to all vertices in G)
E(T) ← ∅
while There is some X ∈ V(T) such that |X| ≥ 2 do
  Let s, t be any two distinct vertices in X
  Let C_1, C_2, ..., C_k be the connected components of T − X
  S_i ← ∪_{Y∈C_i} Y for each 1 ≤ i ≤ k (i.e. all vertices of V represented in some vertex in C_i)
  Let H be the graph obtained from G by contracting each S_i to a single vertex v_{S_i}
  Find a minimum s − t cut S in H and let A = X ∩ S, B = X − S.
  V(T) ← (V(T) − {X}) ∪ {A, B}
  for each edge e = XY ∈ E(T) incident with X do
    let i be such that Y ∈ C_i
    if v_{S_i} ∈ S then
      e' ← AY
    else
      e' ← BY
    end if
    E(T) ← (E(T) − {e}) ∪ {e'}
    w(e') ← w(e)
  end for
  E(T) ← E(T) ∪ {AB}
  w(AB) ← μ(δ(S)) {The capacity of δ(S) in H}
end while
Replace all {v} ∈ V(T) by v and all {u}{v} ∈ E(T) by uv.
Return (V(T); E(T)).
```

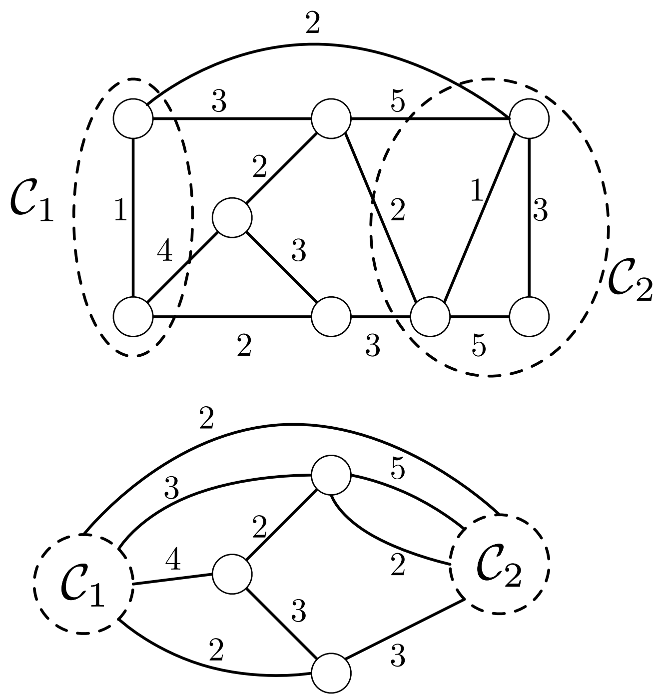

Figure 6.1: Above: a graph $\mathcal{G}$ and two highlighed subsets of nodes $\mathcal{C}_1, \mathcal{C}_2$. Below: the graph $\mathcal{G}/\mathcal{C}_1, \mathcal{C}_2$ obtained by contracting the two subsets to single nodes. Note we keep parallel edges that result from this contraction, but any edge fully contained in a single connected component is deleted.

Note that the vertices of the intermediate trees $\mathcal{T}$ will be vertex sets of the original graph. Indeed they formed a partition of $V(\mathcal{G})$ throughout the algorithm: At the beginning, the only vertex of $\mathcal{T}$ is $V(\mathcal{G})$. In each iteration, a vertex of $\mathcal{T}$ containing at least two vertices of $\mathcal{G}$ is chosen and split into two.

6.2 Proof of Correctness

To prove the correctness of this algorithm, we establish the following loop invariant. For brevity, for any edge $XY \in V(\mathcal{T})$ at any point of the algorithm we instead let C'_{XY} refer to the set of vertices contained in some Z that is connected to X after removing XY from $E(\mathcal{T})$ (this is very similar to the fundamental cut, but not quite the same as, technically, the fundamental cut would be a subset of $V(\mathcal{T})$, which is a collection of subsets of V).

Lemma 3 *Initially and after each iteration, for any edge $YZ \in E(\mathcal{T})$ there is some $s \in Y, t \in Z$ such the fundamental cut C'_{YZ} is a minimum s,t cut (i.e. $w(YZ) = \lambda_{s,t}$).*

Proof. The statement is trivial at the beginning of the algorithm when $\mathcal{T}$ contains no edges; we show that it is never violated during the iterations. So let fix X, s, t, S, A, B for an iteration of the algorithm which before starting it, above statement holds. By renaming s and t if necessary we assume $s \in S$ (thus $s \in A$).

We first show the invariant holds for the new edge AB.

Claim 1 *Expanding the contracted nodes v_{S_i} lying in S is a minum $s-t$ cut in $\mathcal{G}$.*

Note, the set mentioned in the claim is simply C'_{AB} at the end of this iteration. Since $s \in A, y \in B$ this shows the invariant holds for the new edge AB.

Proof. Let us contract $S_1, S_2, ..., S_k$ (connected components of $\mathcal{T} - X$) one by one; for $0 \leq i \leq k$ let $\mathcal{H}_i$ be the graph arise from $\mathcal{G}$ by contracting each of $S_1, S_2, ..., S_k$ to a single vertex. Note $\mathcal{H}_k$ was called $\mathcal{H}$ in the algorithm and $\mathcal{H}_0$ is the original graph $\mathcal{G}$. A straightforward proof by induction that uses Lemma 2 and the fact that each S_i is a minimum $s_i - t_i$ cut for some pair (by the loop invariant) shows that $\mathcal{H}_i$ contains an $s-t$ cut with capacity equal to the minimum $s-t$ cut capacity in $\mathcal{G}$. So expanding the contracted nodes in the minimum $s-t$ cut S computed in $\mathcal{H}$ yields a minimum $s-t$ cut in $\mathcal{G}$. ∎

Next we examine edges that were of the form XY but were replaced by AY or BY. We suppose XY is replaced by AY, the other case is proven in an identical way. Note that by the invariant, there is some $p \in X, q \in Y$ such that C'_{XY} (before modifying $\mathcal{T}$) is a minimum $p-q$ cut in $\mathcal{G}$. If $p \in A$ and because the set of nodes in C'_{AY} (after the modification) is the same as the set C'_{XY} (before the modification) shows the invariant continues to hold for edge AY. So, suppose $p \in B$.

Claim 2 $\lambda_{s,q} = \lambda_{p,q}$

This would complete the analysis of this case, as we then would have that $\mu(\delta_{\mathcal{G}}(C'_{AY})) = \lambda_{p,q} = \lambda_{s,q}$ and $s \in A, q \in Y$.

Proof.

From Lemma 1 we have:

$$\lambda_{s,q} \geq \min\{\lambda_{s,t}, \lambda_{t,p}, \lambda_{p,q}\}. \tag{6.2}$$

For brevity let $\overline{S} \subseteq V$ be the set obtained from S by expanding the contracted nodes in S.

Claim 1 shows $\delta(\overline{S})$ is a minimum $s-t$ cut. Also, because $s, q \in \overline{S}$ we can conclude from Lemma 2 that some minimum $s-q$ cut in $\overline{G}$ is disjoint (or contains) $V - \overline{S}$. Because $t, p \in B \subseteq V - \overline{S}$, this means that adding an edge tp with arbitrarily high capacity does not change $\lambda_{s,q}$. Hence:

$$\lambda_{s,q} \geq \min\{\lambda_{s,t}, \lambda_{p,q}\}. \tag{6.3}$$

Also, because the minimum $s-t$ cut $\overline{S}$ also separates p and q we have $\lambda_{s,t} \geq \lambda_{p,q}$ meaning

$$\lambda_{s,q} \geq \lambda_{p,q}. \tag{6.4}$$

To prove equality observe that $w(XY)$ is the capacity of a cut s and q. Hence:

$$\lambda_{s,q} \leq w(XY) = \lambda_{p,q} \Rightarrow \lambda_{s,q} = \lambda_{p,q}. \tag{6.5}$$

This completes the proof of the second claim. ∎

Finally, every other edge $YZ \in E(\mathcal{T})$ at the start of the iteration where $Y, Z \neq X$ is not altered and C'_{YZ} remains unchanged after the iteration, so the claim continues to hold for them. ∎

Now we are ready to provide the final theorem which is the correctness of the algorithm.

Theorem 1 (Gomory and Hu [1961]) *Every undirected graph possesses a Gomory-Hu tree, and such a tree is found in $O(n^3\sqrt{m})$ time.*

Proof. It is fairly straightforward to see the complexity of the algorithm is dominated by $n-1$ times the complexity of finding a minimum $s-t$ cut. By assuming that we can find a minimum cut in $O(n^2\sqrt{m})$, we obtain the $O(n^3\sqrt{m})$ for the algorithm.

The fact that the invariant Lemma 3 holds at the end of the last iteration and the fact that every vertex in $V(\mathcal{T})$ (just before the final conversion of nodes in $\mathcal{T}$ from singleton sets to vertices in V) is a singleton set shows that the fundamental cut of every edge $uv \in E(\mathcal{T})$ (just after the conversion) is a minimum $u-v$ cut in $\mathcal{G}$. ∎