CSE341T/CSE549T **11/17/2014**

Maximum Flow

Lecture 22

1 Definitions

We will now see a parallel maximum flow algorithm. Flow algorithms are used for various applications like logistics, routing, etc. We will study the basic max-flow algorithm.

Definition 1 *A* flow network *is a directed graph* $G = (V, E)$ *with* source $s \in V$ *and* sink $t \in V$ *and each edge* (u, v) *has a* capacity $c(u, v)$.

We will make some assumptions about the flow network. First, we assume that if there is an edge (u, v), then there is no reverse edge (v, u). (We can remove this assumption by adding an intermediate vertex w between v and u and adding an edges (v, w) and (w, u) with the same capacity as (v, u).) We also assume that there are no self loops. Finally, we assume that there is a path from s to every vertex v and from every vertex v to t. If this is not true, we can simply remove the vertices in question.

We define valid flow f such that it satisfies the following conditions:

1. Capacity Constraint: For all edges (u, v), $0 \leq f(u, v) \leq c(u, v)$
2. Flow conservation: For all vertices $u \in V - \{s, t\}$, $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$.

Our goal is to compute the maximum flow, that is the flow such that $\sum_{u \in V} f(s, u)$ is maximum among all valid flows.

Given some flow along some edges, we can define a residual graph edges E_R as follows: residual $r(u, v) = c(u, v) - f(u, v)$ if $(u, v) \in E$ and $r(u, v) = f(v, u)$ if $(v, u) \in E$; an edge (u, v) is in E_R if $r(u, v) > 0$. The residual graph $G_R = (V, E_R)$. The residual graph indicates how much additional flow can be pushed along each edge.

Claim 1 *If f is a max-flow, then s and t are disconnected in G_R. Conversely, if f is any valid flow and s and t are disconnected in G_R, then f is the max-flow.*

2 Preflow-Push Algorithm

We first define a pre-flow: A pre-flow is similar to a flow, but it doesn't follow the flow conservation property. For each node, we define an excess $e(v) = \sum_{u \in V} f(u, v) - \sum_{u \in V} f(v, u)$. In a valid preflow, $e(v) \geq 0$.

In addition, in preflow-push algorithms, each node v has a label $h(v)$ where $h(s) = n$; $h(t) = 0$ and for all nodes, $h(v) \leq h(u) + 1$ if edge $(v, u) \in E_R$. That is, a node's label is no more than 1 bigger than all its neighbors in the residual graph.

Initially, all labels except the source are 0 and they change as the algorithm proceeds. However, the labels should always remain valid.

2.1 Intuition Behind the Algorithm

We can think of all edges in the residual graph as pipes and vertices as pipe junctions. Each vertex has a large reservoir which holds the excess at the vertex. It sits on a hydraulic platform whose height increases as the algorithm proceeds (height is the label). The flow always flows from a vertex at a higher height to a vertex at a lower height.

2.2 The Algorithm

We first initialize the flow by setting all the labels etc.

```
1  for each v ∈ V
2       do h(v) ← 0
3          e(v) ← 0
4  for each edge (u, v) ∈ E
5       do f(u, v) ← 0
6  h(s) ← n
7  for each vertex v ∈ Adj(s)
8       do f(s, v) ← c(s, v)
9          e(v) ← c(s, v)
```

Then we have two available functions: Push and Relabel.

PUSH(u, v)

1 $\triangleright$ Applies when: $e(u) > 0$, $r(u, v) > 0$, and $h(u) = h(v) + 1$.
2 $\triangleright$ Action: Pushes $\Delta(u, v) = \min\{e(u), r(u, v)\}$ flow from u to v
3 **if** $(u, v) \in E$
4 **then** $f(u, v) \leftarrow f(u, v) + \Delta(u, v)$
5 **else** $f(v, u) \leftarrow f(v, u) - \Delta(u, v)$
6 $e(u) \leftarrow e(u) - \Delta(u, v)$
7 $e(v) = e(v) + \Delta(u, v)$

RELABEL(u)

1 $\triangleright$ Applies when: $e(u) > 0$, for all $v \in V$ s.t. $(u, v) \in E_R$, we have $h(u) \leq h(v)$.
2 $\triangleright$ Action: Increase the height of u so it can push flow.
3 $h(u) \leftarrow 1 + \min_{(u,v)\in E_R}\{h(v)\}$

The push relabel algorithm basically applies push and relabel to edges and vertices until neither applies.

2.3 Proof of Correctness

Lemma 2 *If there is any vertex with excess flow ($e(v) > 0$), then we can always apply either push or relabel.*

Proof. Any vertex v with $e(v) > 0$ always has at least one neighbor in the residual graph. If all the neighbors have the same or larger label, then we can apply relabel. Otherwise, we can push to the neighbor which has the smaller label.

Claim 3 *Applying push and relabel to vertices and edges where they apply always generates a valid preflow and valid labels.*

Lemma 4 *The label of a vertex never decreases. Every relabel operation increases the label by at least 1.*

Lemma 5 *For any valid preflow and valid labeling, s and t are disconnected in G_R.*

Proof. We will prove by contradiction. Say that there is a simple path from s to t where $s(= v_0) \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_{k-1} \rightarrow t(= v_k)$. We know that $k < n$. Also $h(v_i) \leq h(v_{i+1}) + 1$ for all v_i since $(v_i, v_{i+1}) \in E_R$. Therefore, $h(s) \leq h(t) + k \leq 0 + k$, which is a contradiction.

Lemma 6 *If all $e(v) = 0$, then the current preflow is a max-flow.*

Proof. By definition, if all $e(v) = 0$, then the preflow is a flow. Also, for all valid preflows and valid labels, we know that s and t are disconnected in G_R. Therefore, it is also a max-flow.

Lemma 7 *For all v, $h(v) \leq 2n - 1$ at all times.*

Proof. Trivially true for s and t since their labels never change. We only ever change the labels of vertices with excess flow. Suppose $v \in V - \{s, t\}$ has $e(v) > 0$. There must be a path from v to s in G_R, say $v(= v_0) \to v_1 \to v_2 \to ... \to v_{k-1} \to s(= v_k)$. We know that $k < n$. In order to have a valid label, $h(v_i) \leq h(v_{i+1}) + 1$. Therefore, $h(v) \leq h(s) + k \leq 2n - 1$.

Lemma 8 *The algorithm terminates since there are at most $O(n^2)$ relabel operations — once all vertices reach the maximum possible height, there can be no more push or relabel operations and the algorithm terminates finding the max flow.*

2.4 Parallel Push-Relabel Algorithm

```
1  Initialize
2  Repeat until done:
3      do In parallel, push on all edges you can push on.
4         In parallel, relabel all vertices that can be relabeled.
```

We have to be careful while doing the push in parallel — for each vertex with $e(v) > 0$, do a sum-scan on all the outgoing edges with $r(u, v) > 0$ and binary search for $e(v)$ on the result to find all the edges to push on. Therefore, each round takes $O(m)$ work and $O(\lg n)$ span.

Theorem 9 *The total number of rounds is $O(n^2)$.*

Proof. We define $\Phi = \max\{h(v) : e(v) > 0\}$ and $L = \sum_{v \in V} h(v)$. Note first that L always increases since all labels always increase. Since Φ is only defined over vertices with excess flow, it can increase and decrease. We consider two cases:

1. Some labels changed during the round: L increased by at least 1.
2. No labels changed during the round: Vertices with excess flow must have pushed all their excess flow to the neighbors, since otherwise they would have been relabeled. Therefore, Φ decreased by 1, since their neighbors now have excess flow.

There are at most $O(n^2)$ rounds of the first type, since L only increases $O(n^2)$ times. In addition, at the beginning, $\Phi = 0$ and at the end $\Phi = 0$. Φ can only increase in a round if L also increases, therefore, the number of times Φ increases is $O(n^2)$. Therefore, the number of times Φ can decrease is also $O(n^2)$. Therefore, there are at most $O(n^2)$ rounds of the second type.

Therefore, the total work of our max-flow algorithm is $O(n^2 m)$ and the span is $O(n^2 \lg n)$.