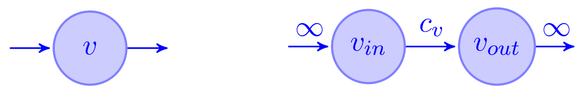
CMPS102: Introduction to Analysis of Algorithms
Prof. Abhradeep Guha Thakurta
University of California Santa Cruz
June 7, 2017

1. (**Node-Capacitated Networks**)

   (a) **Algorithm** Given a node-capacitated network of cities G with source s and sink t, we construct an edge-capacitated graph G' as follows. Each node v in G, besides s and t, is transformed to two nodes v_{in} and v_{out} in G'. All incoming edges to v are replaced with edges pointing to v_{in} while all outgoing edges from v are replaced with edges leaving v_{out}. These edges have infinite capacities. G' also has an edge from v_{in} to v_{out} with capacity c_v.

   (a) a node v in G (b) nodes v_{in} and v_{out} in G'

   We compute a max flow f' from s to t in G' by running a max flow algorithm. Once f' is obtained, we transform it to a max flow f in G by contracting all edges of the form (v_{in}, v_{out}) to v.

   Correctness The flow f in G satisfies flow conservation because the flow f' in G' does. It satisfies the node-capacity constraints in G because f' satisfies the edge-capacity constraints in G'. The flow value (the amount of flow coming out of s) is the same for f and f'. So, the value of f is equal to the value of max flow in G'.

   The value of max flow in G cannot be larger than the value of max flow in G' because any flow in G can be be converted to a flow of the same value in G'. For every node v, other than the source and the sink in G, the flow going through v in G can pass through the edge (v_{in}, v_{out}) in G'. It will not exceed the capacity of this edge because the capacity of v in G was not exceeded.

   Time and Space Complexity Graph G' has $O(m + n)$ edges and $O(n)$ vertices, and can be generated in $O(m + n)$ time and space. Infinite capacities can be modeled by choosing a number C larger than the sum of all node capacities. The capacity-scaling algorithm will do better than Bellman-Ford because Bellman-Ford has running time proportional to C, while the capacity-scaling has running time proportional to $\log C$. It is even better to use the max flow algorithms that runs in $O(|V||E| \log |V|)$ time. Employing this algorithm to find a max flow in G' takes $O((m+n)n \log n)$ time. It takes $O(m + n)$ time to convert f' to f. Hence, the running time is $O((m + n)n \log n)$.

   It takes $O(m + n)$ space to store a graph and run a max flow algorithm.

   (b) By the max-flow min-cut theorem, the max flow value in G' equals the capacity of its minimum cut. Only the edges of the form (v_{in}, v_{out}) can be in this cut, since edges of the form of (v_{out}, v'_{in}) have infinite capacities. In the original node-capacitated graph G, this corresponds to deleting a subset of the nodes. The capacity of such subset is defined as the sum of capacities of deleted nodes.

   (c) By the explanation in part (b), in a node-capacitated network, the max flow value equals the minimum capacity of a set of nodes whose deleting disconnects the source and the sink.