Multicommodity Flow

Q: Given a directed graph $G=(V,E)$ with capacities $c(u,v)$ and a set of $K$ commodities, each with a source, sink, and demand $(s_i, t_i, d_i)$, determine whether flow exists.

Example:

Cars & Trucks

Can we route 2 cars, 1 truck? No.

Solution? No poly-time combinatorial algorithm is known! But linear programming gives a poly-time solution.

LP solution:

Create a variable $f_i(u,v)$ for each edge $(u,v) \in E$, and each commodity $i$.

Then we want to minimize $0 \leftarrow$ (just looking for feasible solution)

subject to $\sum_{i=1}^{K} f_i(u,v) \leq c(u,v)$ $\forall (u,v) \in E$

$\sum_{v \in \text{st}(u)} f_i(u,v) = \sum_{v \in \text{st}(u)} f_i(x,v)$ $\forall i$, $\forall x \in V - \{s_i, t_i\}$

$\sum_{v \in \text{st}(s_i)} f_i(s_i,v) = d_i$ $\forall i$

$f_i(u,v) \geq 0$ $\forall i$, $(u,v) \in E$

That's it!

Analysis: How many variables? $K \cdot E$

How many constraints? $O(K \cdot E)$

both polynomial

Q: What if we add costs $q_i(u,v)$ for routing one unit of good $i$ on edge $(u,v)$?
How to Solve LPs?

- In class, we saw alg for 2-dimensions/variables, n constraints, in expected $O(n)$ time

- Most famous algorithm for general case is simplex algorithm (Dantzig '47).
- Feasible region is a polytope in d dimensions.
- Optimal solution must always exist on an extreme point (i.e. ver of the polytope, unless solution is unbounded.

- Simplex alg starts at some vertex, walks to neighboring vertices until an optimal point is found.
- Different variants choose neighboring vertices differently (called “pivot rule”)

Simplex running Time?

Worst case: How many vertices are there?

With n constraints in d dimensions, $\binom{n}{d}$ \( \in O(2^n) \)
Exponential if \( n \) is \( O(d) \)

Open questions:

1) Does there exist a good pivot rule that only checks $\text{poly}(n,d)$ vertices?

2) Does there even always exist a path to optimal vertex of length $\text{poly}(n,d)$? **Hirsch Conjecture**

Other algorithms: “Ellipsoid,” “Interior Point” are weakly polynomial, i.e., running time is polynomial in \( n, d \), and the number of bits it takes to represent the coefficients of the LP.

Open question: 3) Is there a strongly polynomial time LP solver?