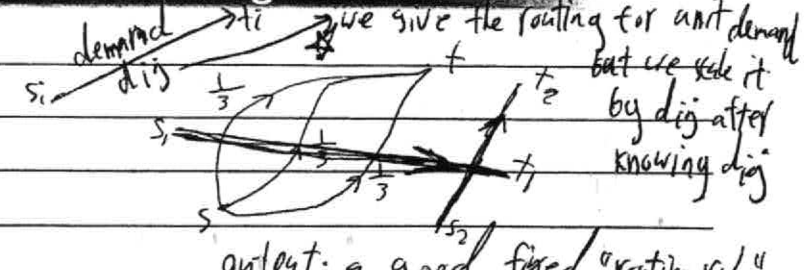
Oblivious Routing

Input:
- (wireless) Network $G(V,E)$
- Source target pairs (s_i, t_i) of G
- For every source/target pair (s_i, t_i), a demand d_i



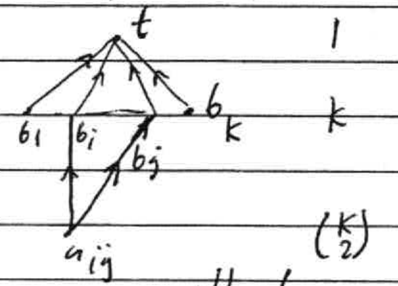
output: a good fixed "routing rule" for every source/target pair

Goal: minimize congestion $\left(\frac{\text{flow}}{\max \text{capacity}}\right)$

Goal: The fixed routing rule should obtain close to optimum congestion on any set of demands. i.e.

$$\text{Minimize} \max_{\text{demand-matrix } D} \left\{ \frac{\text{congestion}_{OBL}(D)}{\text{congestion}_{OPT}(D)} \right\}$$

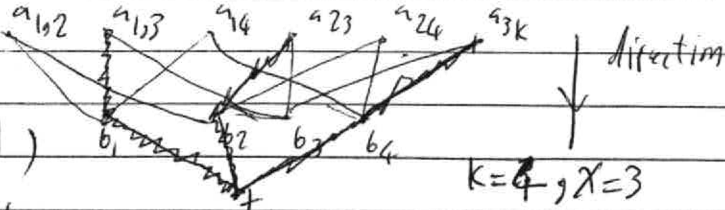
- For undirected graphs there are oblivious Routing Algorithms with competitive ratio $O(\log^3 n)$ [Räcke, FOCS'02], $O(\log^2 n \log\log n)$ [Harrelson, Hildrum, Rao, SPAA'03] and finally optimum solution $\Theta(\log n)$ by [Räcke, STOC'08].
- For directed or even node weighted graphs, there are directed graphs in which every oblivious routing algorithm has competitive ratio in $\Omega(\sqrt{n})$ [ACFKR, STOC'03] [HKRL, SODA'05]



Any oblivious routing scheme defines a unit flow from each node a_{ij}. Hence there is a node b_x which receives at least $\binom{k}{2}/k$ units of flows from its direct neighbors on the first level (i.e., nodes a_{ix} for $i<x$ & a_{xj} for $j>x$).

All edges are unit-capacity

Suppose all neighbors of b_x send a demand of one and all other demands are zero. Then the oblivious algorithm has congestion at least $\binom{k}{2}/k$ at node b_x. However optimum can route this demand with congestion 1 by using the paths $a_{ix} - b_i - t$ for demands from nodes a_{ix} and the paths $a_{xj} - b_j - t$ for demands a_{xj}.
The proof follows immediately since $k = \Omega(\sqrt{n})$.



Important example) $k=4, \chi=3$

Exactly the same construction works for node-weighted graphs (all capacities are one except t with ∞ cap) but this is not a bad case for undirected graphs.

- Though the ratios for directed graphs are $\Omega(\sqrt{n})$ and $\Omega(\lg n)$ for general undirected graphs, but there is a polynomial time construction that gives the true optimal ratio for any (directed/undirected) network. [ACEKR'03]
- If demands are random from known distributions, the competitive ratio is in $O(\lg^2 n)$ with high probability in directed graphs.
- Randomness does not help for undirected graphs (we have lower bound $\Theta(\frac{\lg n}{\lg\lg n})$ in this case.
- Other goals such as maximizing throughput instead of minimizing congestion has also been considered [Awerbuch, H., Kleinberg, Leighton, SODA'05] or oblivious network design with general edge-functions [Gupta, H, Raecke, SODA'06] as we discussed before.

☆ Racke (FOCS 02) introduced a tree decomposition that aims at constructing a tree that does not approximate point-to-point distances in the input graph (like Bartal or FRT'03 technique) but instead approximates the cut structure of the graph in the following sense: Given a concurrent multicommodity flow problem (CMCF-problem in which demands d_{st} from s to t should be shipped simultaneously with minimum congestion (the maximum over all edges of the flow along the edge divided by the capacity of the edge)) in graph G, the optimum solution of the corresponding flow problem in a decomposition tree has a lower congestion; and conversely given a CMCF-problem in a decomposition tree the corresponding problem in G can be solved with a congestion that is only a factor f larger (and it is constructable). These kinds of decomposition is called "cut-decomposition". ($f = O(\lg n)$)

☆ The Model: G with nodes V where $|V|=n$ is given, we have a capacity/weight function c on the edges, where $c(u,v)=c(v,u)$ since the graph is undirected. Assume $c(u,v)>0$ (if there is an edge from u to v) and $c(u,v)=0$ iff $(u,v)\notin E$

(like Bartal or FRT)
Decomposition Trees: A decomposition tree for the graph G is a rooted tree $T=(V_t, E_t)$ whose leaf nodes correspond to nodes in G, i.e., there is a one-to-one relation between nodes in G and leaf nodes in T. Whenever we use the concept of a decomposition tree for a graph G, we implicitly assume that we are also given an embedding of T into G. This means there is a node mapping function $m_V : V_t \to V$ that maps tree nodes to nodes in the graph and because of the property of a decomposition tree this mapping function induces a bijection between leaf nodes of T and nodes in G. We are also give a function $m_E : E_t \to E^*$ that maps an edge $e_t = (u_t, v_t)$ of T to a path P_{uv} between the corresponding end points $u = m_V(u_t)$ and $v = m_V(v_t)$. We also introduce functions $m'_V : V \to V_t$ and $m'_E : E \to E_t^*$ responsible for mapping from G to T. m'_V outputs for a node $v \in V$ the leaf node in T corresponding to v, and the function m'_E gives for an edge $e = (u, v) \in E$ the (unique) shortest path in T between $m'_V(u)$ and $m'_V(u)$.

For a multicommodity flow f_T on a decomposition tree we use $m(f_T)$ to denote the multicommodity flow f_T on a decomposition tree we use $m(f_T)$ to denote the multicommodity flow that is obtained by mapping f_T to G via the edge-mapping function m_E. Similarly, we define for a flow f in G, $m'(f)$ as the flow in T obtained by mapping f to T. Note that we can add flows and scale them e.g. $m(\alpha \cdot f_T + \alpha' f'_T) = \alpha \cdot m(f_T) + \alpha' \cdot m(f'_T)$.

Given a decomposition tree T for G we define the capacity $c(u_t, v_t)$ of a tree edge $e_t = (u_t, v_t)$ as $c(u_t, v_t) := \sum_{u \in V_{u_t}, v \in V_{v_t}} c(u, v)$, where V_{ut} and V_{vt} denote the two partitions of V induced by the cut corresponding to edge e_T.
Given a multicommodity flow in G we want to compare its congestion in G, to its congestion in T.

Thm 1: Suppose you are given a multicommodity flow f in G with congestion C_G. Then the flow $m'(f)$ obtained by mapping f to some decomposition tree T results in a flow in T that has congestion $C_T \le C_G$.
Pf: Suppose an edge $e_t = (u_t, v_t)$ in the tree has congestion C_T. All traffic that traverse the cut in G between V_{ut}, V_{vt}. The total capacity of all edges over this cut is exactly $c(e_t)$. Hence by a simple averaging argument, one of these edges must have relative load at least C_T; this gives $C_T \le C_G$.

Given a decomposition tree with an embedding of this tree into graph G we can ask for the load that is induced on a graph edge e by this embedding. Let

$$\underbrace{\text{load}_T(e)}_{\text{absolute}} := \sum_{e_t \in E_T : e \in m_E(e_t)} c(e_t) \quad \text{and} \quad \underbrace{\text{rload}_T(e)}_{\text{relative load.}} = \frac{\text{load}_T(e)}{c(e)}$$

We are looking for a convex combination of decomposition trees (such as Bartal or FRT) such that for every edge the expected relative load is small, i.e.

$$\text{minimize } \beta = \max_{e \in E} \left\{ \sum_i \lambda_i \, \text{rload}_{T_i}(e) \right\} \text{ where } \lambda_i \geq 0 \text{ and } \sum_i \lambda_i = 1$$

__Thm 2:__ Suppose we are given a convex combination of decomposition trees with maximum expected relative load β and suppose that we are given for each tree T_i a multicommodity flow f_i that has congestion C in T_i. Then the multicommodity flow $\sum_i \lambda_i m_{T_i}(f_i)$ has congestion at most βC when mapped to G. [This is like FRT or Bartal]

__Pf:__ Fix a tree T_i. Routing the flow f_i in the tree generates congestion at most C which means that the amount of traffic that is sent along an edge $e_t = (u_t, v_t)$ is at most $C c(e_t)$. Hence the total traffic that is induced on a graph-edge e when mapping $\lambda_i f_i$ to G is at most $C \lambda_i \, \text{load}_{T_i}(e)$. Therefore the relative load induced on e when mapping all flows $\lambda_i f_i$ is at most $C \sum_i \frac{\text{load}_{T_i}(e)}{c(e)} = C \sum_i \lambda_i \, \text{rload}_{T_i}(e) \leq C\beta$

Räcke '08 shows that we can indeed obtain a convex combination of decomposition trees for which $\beta = O(\log n)$. The proof is not hard but indeed use FRT result and Charikar et al. result (on derandomization of Bartal & FRT). Esp. it uses FRT to obtain $O(\log n)$-approximation for the Minimum Communication cost tree problem. (We omit the details here; see Räcke, STOC '08)

... where j is assigned to $i(j)$ in OPT.

New oblivious Routing Algorithm: The convex combination of decomposition trees defines a unit flow for every source-target pair, by combining for a pair (u,v) the paths between u and v in trees T_i where the path from T_i is weighted with λ_i. This is the oblivious routing

Pf: Now given a demand vector that can be routed with congestion C in G, routing it in a decomposition tree creates congestion less than C in any tree. [Thm] Now mapping the flows from all decomposition trees back (and thus scaling it by a factor λ_i) the thing that indeed we have done in our oblivious routing gives a solution in G with congestion at most $\beta \cdot \max_i \{C_{opt}(T_i)\} \leq \beta \cdot C_{opt}(G)$ due to Thm 2. (note that because of uniqueness of paths in trees all scaling by vector λ has been done automatically.)

Hence the oblivious routing scheme has competitive ratio $O(\lg n)$ as desired. ∎

This convex combination of trees has several other applications like Bartal/FRT Result, e.g. for min bisection, sparsest cut, multicast Routing, online multicast, etc. gives $O(\lg n)$ approximation)

Universal solutions: given a metric space, it is a total ordering of points of the space such that for any finite subset, the tour which visits these points is the given order is as close as possible to the optimal tour. This is Universal TSP.

Universal solutions are motivated by oblivious routing though they were before.

Universal TSP first considered by Platzman and Bartholdi in JACM'89

For planar graphs and for plane graphs there is a lower bound $\sqrt[6]{\frac{\log n}{\log\log n}}$ (HKL'06)

For general graphs the lower bound is $\Omega(\log n)$ for expander graphs. (GKSS'10)

Upper bound is $O(\log n)$ [GHR'06]

Oblivious network design:

Consider multicommodity (uniform) buy-at-bulk in which there is a non-decreasing monotone subadditive function $f: \mathbb{R}^+ \to \mathbb{R}^+$ for an edge e where $f(b)$ is the minimum cost of cable installation with bandwidth b for an edge e.

Given a set of bandwidth demand pairs, install sufficient capacities with minimum total cost.

The uniform one has $O(\log n)$ approximation by Bartal (see the notes) (Non-uniform was $O(\log^4 n)$). The ~~oblivious~~ version in which we decide the paths before hand has $O(\log^2 n)$ algorithm [GHR'06]

Incremental solutions: given a notion of ordering on solutions of different cardinalities k, we give solutions for all values of k such that the solutions respect the ordering and such that for any k, our solution is close in value to the value of an optimal solution of kardinality k. For example for k-median k-MST (the ordering of n edges $e_1, e_2, \ldots e_{n-1}$ such that for any $k \in [2, n]$ the first $k-1$ edges of the sequence span k vertices including r), k-set cover which asks for a min-cost subcollection of C that cover at least k-elements. (outputs an ordering of sets such that for any $k \in [1, n]$, the minimal prefix of the sequence that covers k elements is a good approximation to k-set cover problem). There is $O(1)$-Incremental solutions for k-median and k-MST and $O(\ln n)$ for k-set cover. [LNRW'06]

Also, the people consider the version that we are given a set$_k$ of important vertices called terminal and we want to find a universal solutions whose competitive ratio only depends on k, e.g. polylog(k) for problems such as k-cut, k-multicut, etc. [Moitra'09]. He considers mainly the cut problems [what about the connectivity problems?]