Game Theory in Computer Networks

Mukarram Bin Tariq
mtariq@cc.gatech.edu

IsyE 6230: Economic Decision Analysis

April 19, 2007

Outline

- Part 1: Incentives for file exchange in P2P network
  - How bit-torrent prevents free-riders' problem

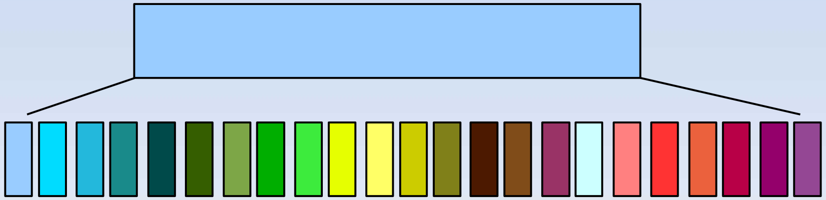
- Part 2: Selfish Routing and Price of Anarchy
  - Pigou and Braess Pradox
  - Relationship of Nash and Optimal Flows

P2P File Exchange Problem

- **Network:** Comprises nodes, seeders and leechers
- **Seeder:** A user that offers a file – has a complete copy of the file – only upload the content
- **Leechers:** A community of users that want to download the offered file – none has a complete copy of the file yet. Leechers both upload and download the content
- **Problem:**
  - Seeder could upload the file individually to each user
    - A lot of work for the seeder
  - Upload to one user and ask it to upload to others
    - Selfish users, go offline after downloading.. free-riders problem
  - *How to have users cooperate, so that everyone can download their complete copy without seeder to have to do a lot of work?* ***<Game Theory Question>***

Setting up Tit-for-Tat

- To play tit-for-tat, there must be some bargaining chips...



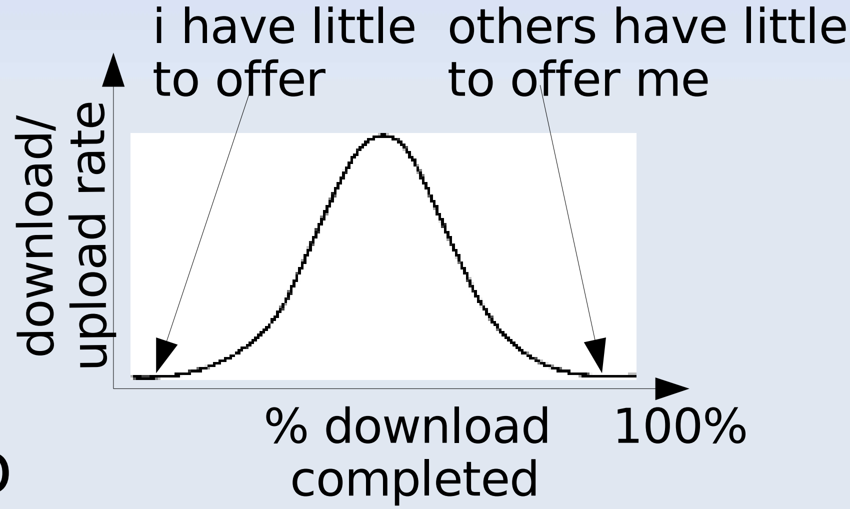
- Seeder creates small chunks of a large file and gives out to leechers
  - Leechers use the chunks as bargaining chips
- Actions:
  - Rarest Piece First:
    - Ensures that pieces spread quickly and users have chips to trade
    - Incentive compatible because speeds up the download

Strategy

- Initially unchoke a few leechers – allow them to bootstrap
- Initial fetching is random (first 4 chunks)
- Repeated game: You give me a chunk, I give you a chunk (tit-for-tat)
  - Note it can only work if the two parties have disjoint sets of chunks – duplicate chunks are worthless
  - RPF promotes diversity in chunk holding

Open Issues

- Last Chunk Problem
  - downloaders have to wait significant time for the last few chunks
- Strange Cooperation Behaviors
  - Can we fix these?
- Clients exploit the boot-strap process by disconnecting and reconnecting repeatedly
- Can we play a transitive tit-for-tat - be part of multiple swarms at a time

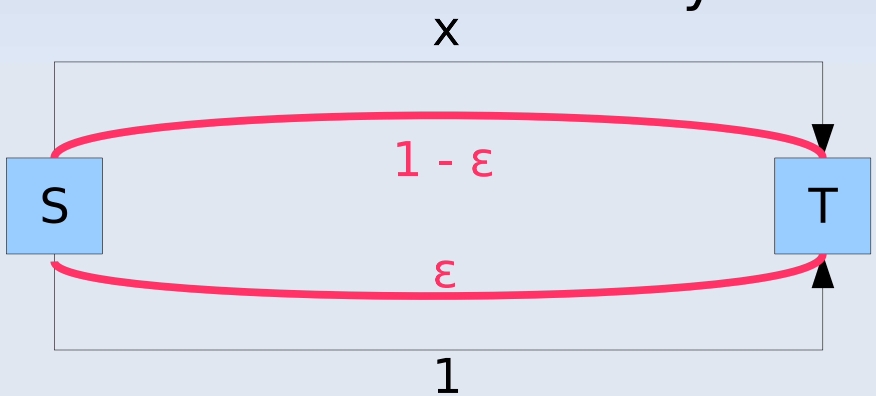


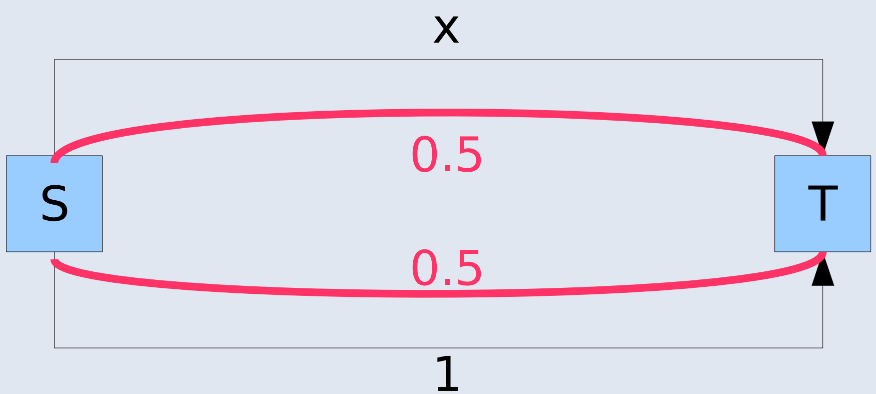
Part II

Selfish Routing

Motivating Example

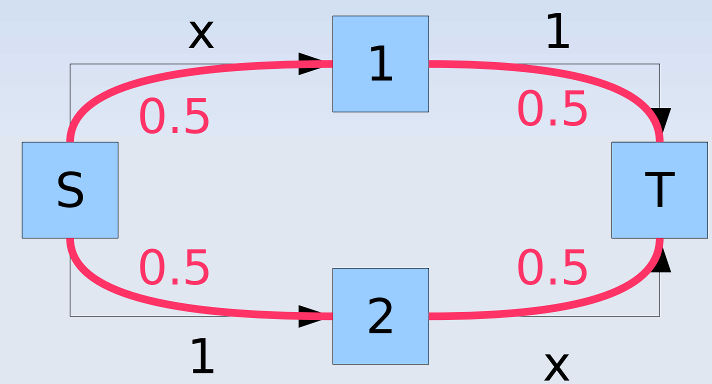
- One unit of traffic is trying to go from S to T
  - Each “packet” wants to reach T with minimum delay
  - Any ε traffic on the lower path feels envy and moves to upper path
- Equilibrium solution: ε = 0
  - Total delay: 1
  - Can we do better?
  - For ε = 0.5, total delay is 0.5 x 1 + 0.5 x 0.5 = 0.75
- Equilibrium solution may not be optimal; ratio 1/0.75 = 4/3

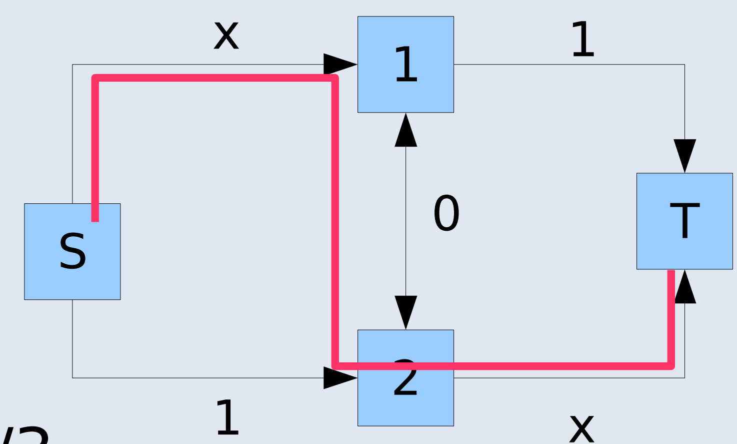




Braess' Paradox

- Can increasing capacity increase the delay?
- Nash Flow:
  - s-1 = 0.5, s-2 = 0.5
  - 1-d = 0.5, 2-d = 0.5
  - Total Latency = (0.5)(0.5)+1(0.5) = 1(0.5)+(0.5)(0.5) = 1.5
  - Lets add the capacity
  - All traffic incurs more latency; total latency = 2
- Price of Anarchy = 2/1.5 = 4/3





Key Results

- Networks with affine function: $l(x) = ax + b$
  - the price of anarchy is bounded by 4/3
  - Pigou's and Braess's are examples of worse case!
- Networks with $l(x) = (u-x)^{-1}$ form of latency (e.g. M/M/1 queues), the price is unbounded
  - Can be bounded if max traffic R_{max} on any edge is <u>less</u> than the min capacity of any edge u_{min}
  - Bound: $(1+ (u_{min}/(u_{min} - R_{max})^{-1/2}) / 2$
  - Tolerable iff network is sufficiently over-provisioned

The Model

- Some Notation:
  - Network is a directed graph G(V,E),
  - $\{s_1, t_1\}\ldots\{s_k, t_k\}$: k-commodities, characterized by src/dest pairs
  - P_i : Set of Simple Paths b/w $\{s_i, t_i\}$, $P^* = \cup P_i$
  - $f: P^* \rightarrow R^+$: overall flow, f^i_e : flow of commodity i on edge e, f_e: flow on edge
  - r_i : demanded traffic rate for commodity i
  - $l_e(.)$: latency function on edge e
  - $l_p(f)$: $\Sigma_{e \in P} l_e(f_e)$: latency of a path
  - $C(f)$: $\Sigma_{P \in P^*} l_P(f) f_P$: Cost of a flow f
  - (G,r,l) : an instance

Nash Equilibrium

- A flow is feasible if $\Sigma_{P \in P_i} f_P = r_i$, i.e. demand is satisfied
- Each unit of a flow travels along the minimum latency path available to it: uses all paths of same latency
- NE or Nash Flow, for all $i \in \{1..k\}$, $p_1, p_2 \in P_i$, we have

$$l_{P1}(f) \leq l_{P2}(\tilde{f})$$

$$\tilde{f}_P = \begin{cases} f_P - \delta & \text{if } P = p_1 \\ f_P + \delta & \text{if } P = p_2 \\ f_P & \text{otherwise} \end{cases}$$

- All this says is that due to greedy and selfish nature, all paths for a commodity have same delay, lets denote that by $L_i(f)$, then the cost of the flow at Nash Equilibrium can be written neatly as: $C(f) = \sum_{i=1}^{k} L_i(f) r_i$

Optimal Flows

$$\min \quad \sum_{e\in E} \{c_e(f_e)=l_e(f_e)f_e\}$$

$$s.t.$$

$$\sum_{p\in P_i} f_p = r_i \qquad \forall i\in\{1,\dots,k\}$$

$$f_e = \sum_{p\in P: e\in E} f_p \qquad \forall e\in E$$

$$f_p \geqslant 0 \qquad \forall p\in P$$

- A flow is optimal if the marginal benefit of decreasing a flow along a path is at most the marginal cost of increasing flow on any other path, i.e., $p_{1,} p_2 \in P_{1,} \quad f_{p_1} > 0,$
- $c'_{p_1}(f) \leqslant c'_{p_2}(f), \forall i\in\{1..k\}$
-

Relationship between Optimal and Nash Flows

- Lets define marginal cost as

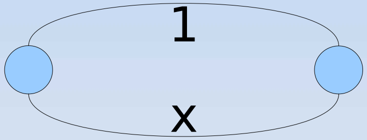
$$l^*_e = \frac{d}{dx}(y l_e(y))(x) = l_e(x) + x l'_e(x)$$

-

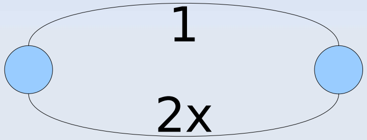
- Then we can see that *(G,r,l)* is optimal iff it is at Nash Eq for the instance *(G,r,l*)*

Examples

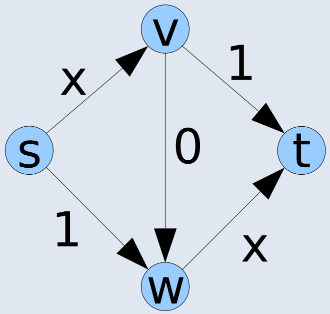
- Pigous Network
  - latency functions are 1 and x
  - Marginal cost functions are 1 and 2x
  - (0.5 0.5) gives non-envious flows (NE) under marginal functions
  - Therefore (0.5 0.5) is the optimal solution
- Braess Network
  - 0.5 on svt and 0.5 on swt makes all three paths equal under marginal cost functions
  - therefore 0.5 on svt and 0.5 on swt is optimal under latency fns



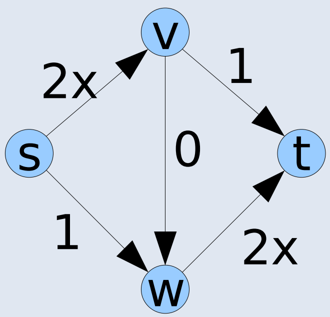
latency fns



marginal cost fns



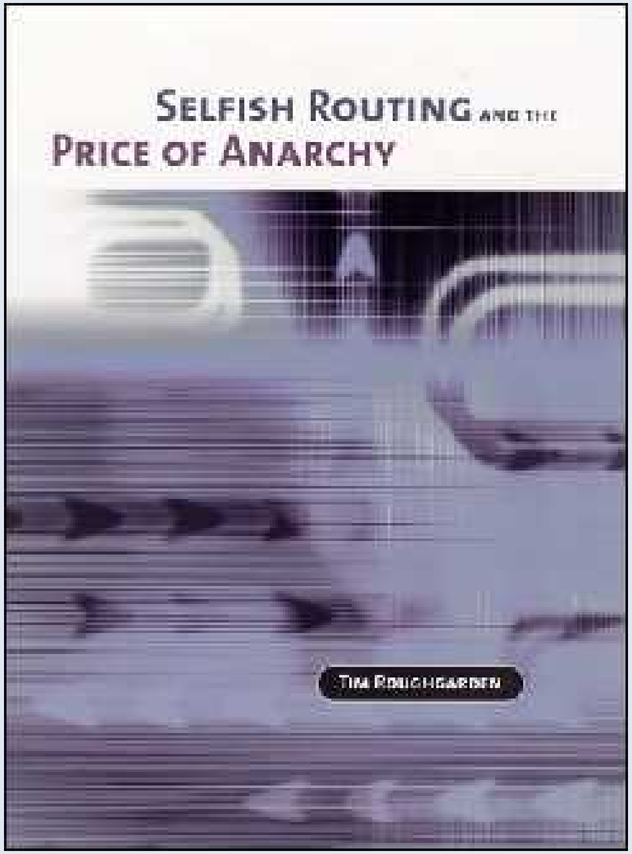
latency fns



marginal cost fns

For more information

- Tim Roughgarden's webpage
  - http://theory.stanford.edu/~tim/
  -
- His Thesis
  - MIT Press
  - ISBN:0-262-18243-2



Some Bittorrent Facts

- The protocol was designed in April 2001, implemented and first released July 2, 2001 by programmer Bram Cohen, and is now maintained by BitTorrent, Inc.
- By some estimates, Bittorrent comprises 55% of access network and 35% core network bandwidth in the Internet today – making it the single largest application
- Some trackers: The pirates bay, Torrent Reactor, Torrent Spy, Mininova, Supernova(RIP)
  - There are some DHT based trackers as well