6.263 Data Communication Networks

Application of Optimal Routing

An Application of Optimal Routing

MATE: Multipath Adaptive Traffic Engineering

Shortest Path vs. Optimal Routing

- Internet routing uses shortest path. Why?
  - Shortest path is simple because it decouples congestion from routing
  - Optimal routing
    - Needs to know traffic demands between all OD pairs r_w
    - Needs to know the paths
    - Paths and r_w should be stable enough
    - Should find a distributed version of the algorithm

3

What's Traffic Engineering?

- ISPs are in the business of mapping traffic demands onto the underlying topology
- Good mapping removes hot spots and congestions
- The job of traffic engineering is to find a good mapping
- The mapping adapts to deal with changing network conditions, i.e., congestion and failures
- How do you do it?

4

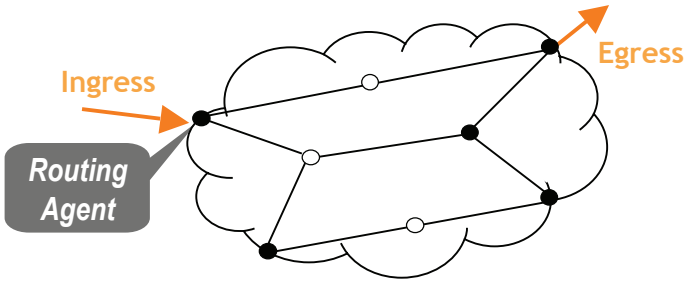
We need to address multiple sub-problems

1. Given an optimal routing X^*p, force the traffic along the desired paths
2. Find the optimal traffic split X^*p, i.e., the optimal routing
3. Split the traffic: Traffic is not fluid, it is packets and flows

5

Sub-Problem 1: Given a balanced traffic split, force the traffic along the desired paths

Solution:



- An optimal routing agent per OD pair, at ingress node
- Agent knows the paths between OD pair, which are computed offline
- Paths are pinned using MPLS tunnels (These are like virtual circuits discussed in lecture 1)

6

Sub-Problem 2: Find the optimal traffic splits X^*_p

Solution:

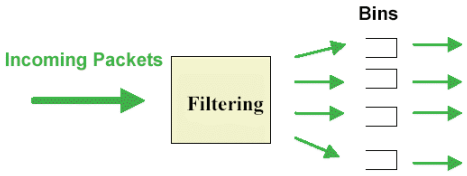
- Define cost as delay in an M/M/1 queue C(Fij)= 1/(Cij-Fij)
- Derivatives need to be measured
  - Periodically send time-stamped probe packets from ingress to egress and measure one-way delay ◦ compute change in delay
- Also can incorporate losses in the cost function as very large delays
- Need to solve the optimization in a distributed way
  - See the MATE paper in your reading list for a distributed approach

7

Sub-Problem 3: Split traffic according to desired X^*_p

Solution:

- Each flow should stick to one path (because TCP reacts badly to packet reordering)
- For each incoming packet, hash (src IP, dst IP, src port, dst port) to some large space ◦ use hash to assign packet to a specific bin
- Assign bins to paths according to X^*_p



- Bins faciliate traffic shifting
- Hashing based on IP addresses
- Packet order for a given flow is maintained

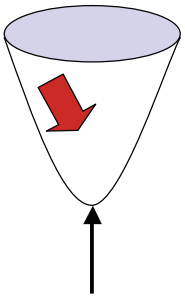
8

Outline

- Gradient and Hessian
- Descent algorithms
- Step size
- Convergence
- Constraints

Gradient and Hessian

Gradient of some function f : $\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x(1)} \\ \dots \\ \frac{\partial f(x)}{\partial x(n)} \end{bmatrix}$



Descent direction

Optimum point

Hessian : $\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x(1)^2} \dots & \frac{\partial^2 f(x)}{\partial x(1)\partial x(n)} \\ \dots & \dots \\ \frac{\partial^2 f(x)}{\partial x(n)\partial x(1)} \dots & \frac{\partial^2 f(x)}{\partial x(n)^2} \end{bmatrix}$

Convex function: the Hessian is positive semidefinite and symmetric

Positive (semi) definiteness

- Matrix A is positive (semi) definite if for any vector $\underline{x}$, $\underline{x}^T A \underline{x} > (\geq) 0$
- A symmetric matrix is positive (semi) definite iff all of its eigenvalues are positive (non-negative)
- A square symmetric positive semi definite matrix has a symmetric square root Q such that $Q^2 = A$
- The inverse of a positive definite symmetric matrix is also symmetric and positive definite (the eigenvalues of the inverse matrix are the inverses of the eigenvalues of the original matrix)

Choice of step

- General set-up

$$x(p)^{k+1} = x(p)^k - \alpha^k B^k \nabla D(x^{k+1})$$

- We may vary the choices of B and α
- Example: steepest descent, $B = I$

$$x(p)^{k+1} = x(p)^k - \alpha^k \nabla D(x^{k+1})$$

All vectors

- Other choices: Newton's method

Choice of step size

- Minimization rule: choose α^k that minimizes along some direction d^k :

$$D(x^k + \alpha^k d^k) = \min_{\alpha \geq 0} D(x^k + \alpha d^k)$$

- Armijo rule: successive stepsize reduction. Select a step size. If the ensuing vector is not an improvement, reduce the step size. Need to do so in a way that is guaranteed to work: fix s, $0< \beta <1$, $0< \sigma <1$, pick m_k to be smallest non-negative m such that

$$D(x^k) - D(x^k + \beta^m s d^k) \geq -\sigma \beta^m s \nabla D(x^k)' d^k$$

select $\alpha^k = \beta^{m_k} s$

MIT

Convergence

- Convergence results exist for many of these systems
- The eigenvalues of the Hessian are often crucial in determining how "different" various directions are
- For a quadratic function, method of steepest descent and α chosen to minimize D at each step

$$\frac{D(x^{k+1}) - D^*}{D(x^k) - D^*} \leq \left(\frac{M - m}{M + m}\right)^2$$

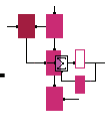
M is the largest eigenvalue of the Hessian

m is the smallest eigenvalue of the Hessian

- For large spread in eigenvalues, a large change in x may translate into small cost difference

MIT

Convergence



- Proof relies on Kantorovich inequality:

$$\frac{(y^T y)^2}{(y^T Q y)(y^T Q^{-1} y)} \leq \frac{4Mm}{(M+m)^2}$$

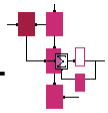
Q positive definite and symmetric

y any non - zero real vector

- Further results: when the direction approaches the Newton method and the Armijo step size rule is followed, for s=1, σ <0.5, then convergence is super linear and for large enough k the step size is unity

MIT

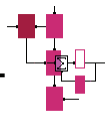
What about our constraints?



- All of these methods are concerned with reducing cost
- However, we must also remain *feasible*
- F-W maintained feasibility by restricting the step size to be of a type that maintained feasibility, at the expense of flexibility and the ensuing speed of convergence
- Can we make use of the more sophisticated step size methods while maintaining feasibility?
- Let us first consider a very simple example: we need to maintain non-negativity

MIT

Optimizing over the positive orthant

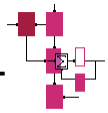


- We consider a steepest descent method with the modification that we project onto the positive orthant

$$x(p)^{k+1} = \left[x(p)^k - \alpha^k \nabla D(x^{k+1})\right]$$

- Most of the methods carry through
- Let us transform our problem into a positive orthant problem
- We use the constraint to turn it into a new problem with no dependence on the MFDL paths

Optimizing over the positive orthant



$$\sum_{(i,j)} D(F(i,j))$$

subject to the constraints

$$F(i,j) = \sum_{\text{all paths p containing } (i,j)} x(p)$$

$$\sum_{p \text{ in } P(w)} x(p) = r(w) \text{ for al OD pairs w in W}$$

$$x(p) \geq 0$$

we create $\tilde{x}$ by expressing the MFDL paths in terms of the other path flows : $x(\bar{p}) = r(w) - \sum_{p \neq \bar{p}} x(p)$

minimize $\tilde{D}(\tilde{x})$

s.t. $\tilde{x}(p) \geq 0$ for p not MDFL

Optimizing over the positive orthant

- When taking derivatives with respect to x(p), all terms with non-MDFL x(p) originally in D remain, but terms corresponding to a MDFL path now have terms in all the non-MDFL path

$$\frac{\partial \tilde{D}(\tilde{x}^k)}{\partial x(p)} = \frac{\partial D(x^k)}{\partial x(p)} - \frac{\partial D(x^k)}{\partial x(\bar{p})} \text{ for non - MDFL p}$$

$$\text{recall } \frac{\partial D(x)}{\partial x(p)} = \sum_{\text{all links (i,j) on path p}} D_{i,j}'\left(\sum_{\text{all paths p containing (i,j)}} x(p)\right) = d_p$$

$$\frac{\partial^2 \tilde{D}(\tilde{x}^k)}{(\partial x(p))^2} = \sum_{\substack{\text{all links (i,j) on path p} \\ \text{or corresponding MDFL,} \\ \text{but not both}}} D_{i,j}''\left(\sum_{\text{all paths p containing (i,j)}} x(p)\right) = H_p$$

Optimizing over the positive orthant

- Use the "second derivative length" in updating

$$x_p^{k+1} = \left\lfloor x_p^k - \alpha^k H_p^{-1}\left(d_p - d_{\bar{p}}\right)\right\rfloor_-$$

- We may use any of the step increase methods that we would normally use
- We may establish convergence results in this way