On the Optimality of Link-State Routing Protocols

Dahai Xu, Ph.D.

Florham Park
AT&T Labs - Research

Joint work with Mung Chiang and Jennifer Rexford (Princeton University)

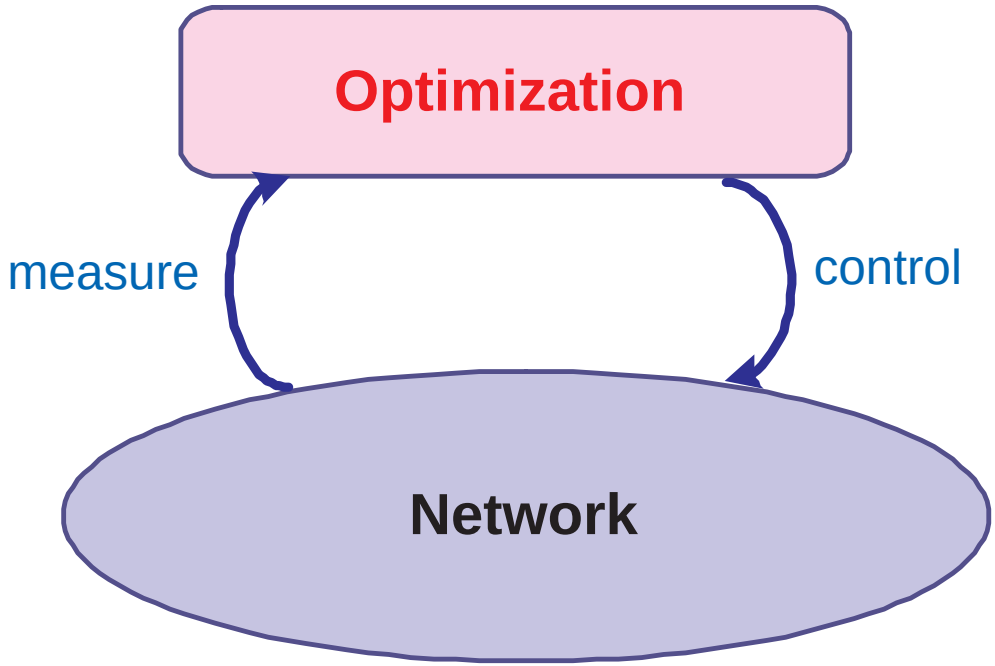
New Mathematical Frontiers in Network Multi-Resolution Analysis
Nov. 18, 2008

Outline

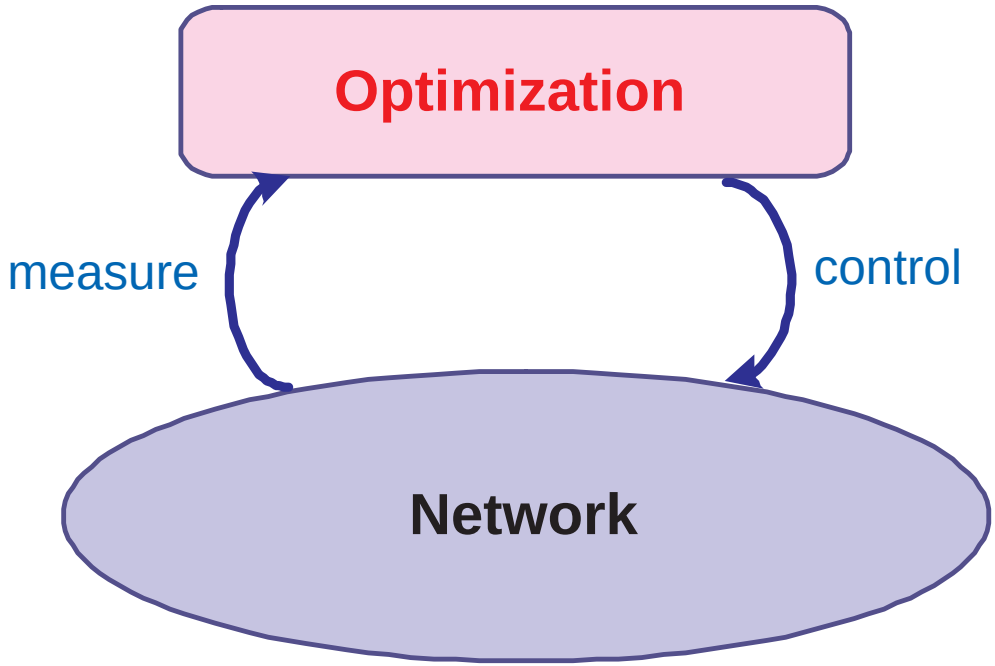
Outline

Outline

Network Management



Network Management



- Settle a long-standing question on network management
- Initiate a new area in IP routing

City Traffic Control

- Big cities suffer from traffic congestion during rush hours
- Transportation administrator: balance traffic by setting price for each road segment

City Traffic Control

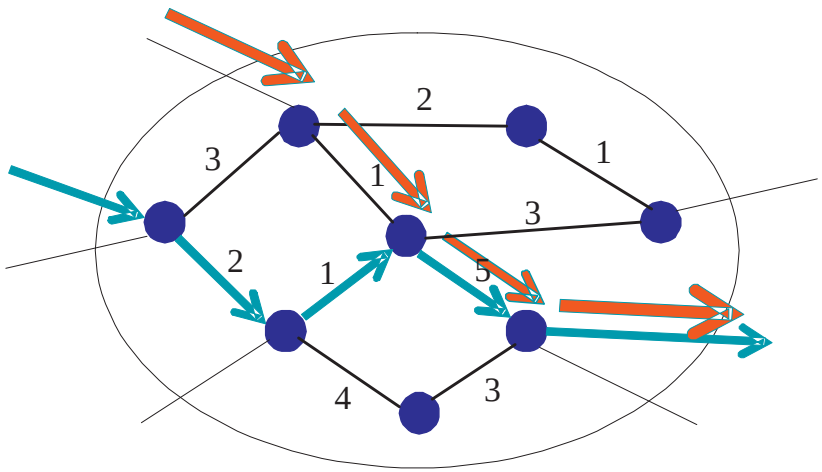
- Big cities suffer from traffic congestion during rush hours
- Transportation administrator: balance traffic by setting price for each road segment
  - Some drivers choose the "cheapest" path
  - Other drivers choose more "expensive" path expecting less congestion (delay)

City Traffic Control

- Big cities suffer from traffic congestion during rush hours
- Transportation administrator: balance traffic by setting price for each road segment
  - Some drivers choose the "cheapest" path
  - Other drivers choose more "expensive" path expecting less congestion (delay)
- How to set road price (link state/weight) $\Rightarrow$ Traffic Engineering

Link-State Routing

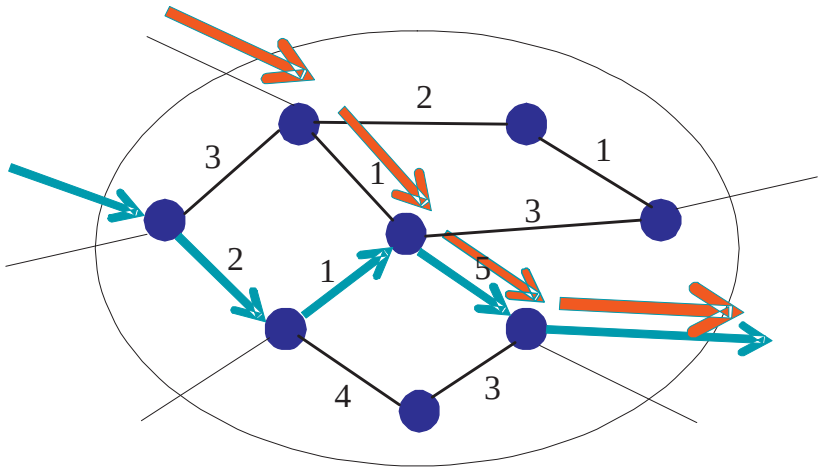
- Routers
  - Exchange link weights (states) with Interior Gateway Protocols (IGPs): e.g. OSPF (Open Shortest Path First)
  - Distributively determine "next hop" to forward a packet
  - Path selection based on link weights (shortest path)



Path length= 8

Link-State Routing

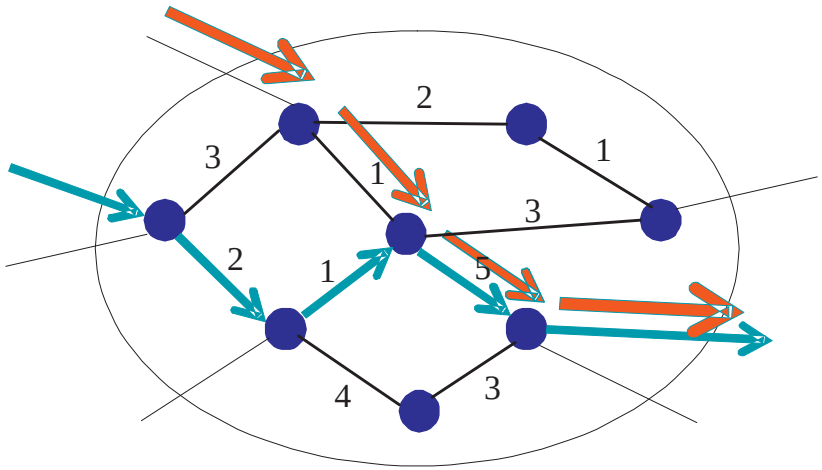
- Routers
  - Exchange link weights (states) with Interior Gateway Protocols (IGPs): e.g. OSPF (Open Shortest Path First)
  - Distributively determine "next hop" to forward a packet
  - Path selection based on link weights (shortest path)
  - Typically splits traffic evenly when multiple shortest paths exist



Path length= 8

Link-State Routing

- Routers
  - Exchange link weights (states) with Interior Gateway Protocols (IGPs): e.g. OSPF (Open Shortest Path First)
  - Distributively determine "next hop" to forward a packet
  - Path selection based on link weights (shortest path)
  - Typically splits traffic evenly when multiple shortest paths exist

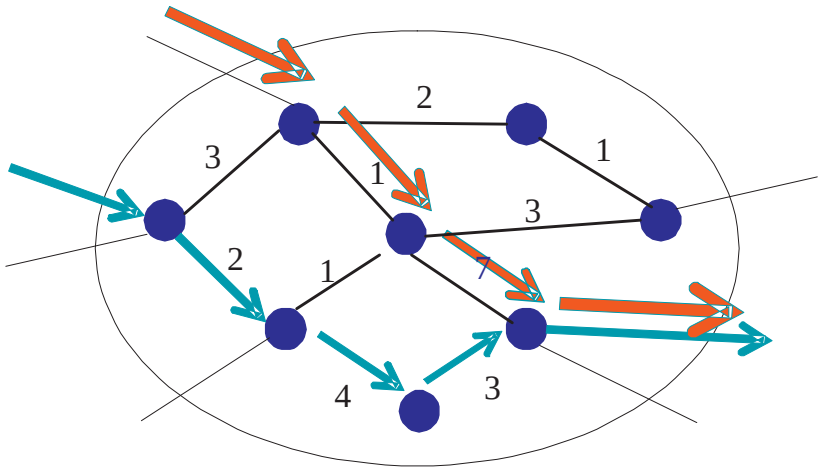


Path length= 8

- Network operator configures link weights to guide routing
  $\Rightarrow$ Traffic Engineering

Link-State Routing

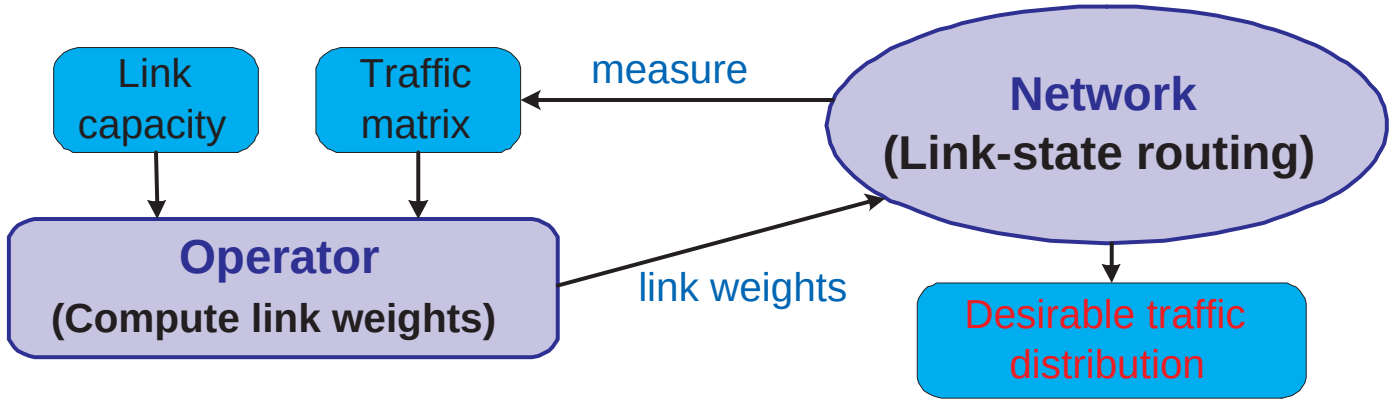
- Routers
  - Exchange link weights (states) with Interior Gateway Protocols (IGPs): e.g. OSPF (Open Shortest Path First)
  - Distributively determine "next hop" to forward a packet
  - Path selection based on link weights (shortest path)
  - Typically splits traffic evenly when multiple shortest paths exist



Path length= 9

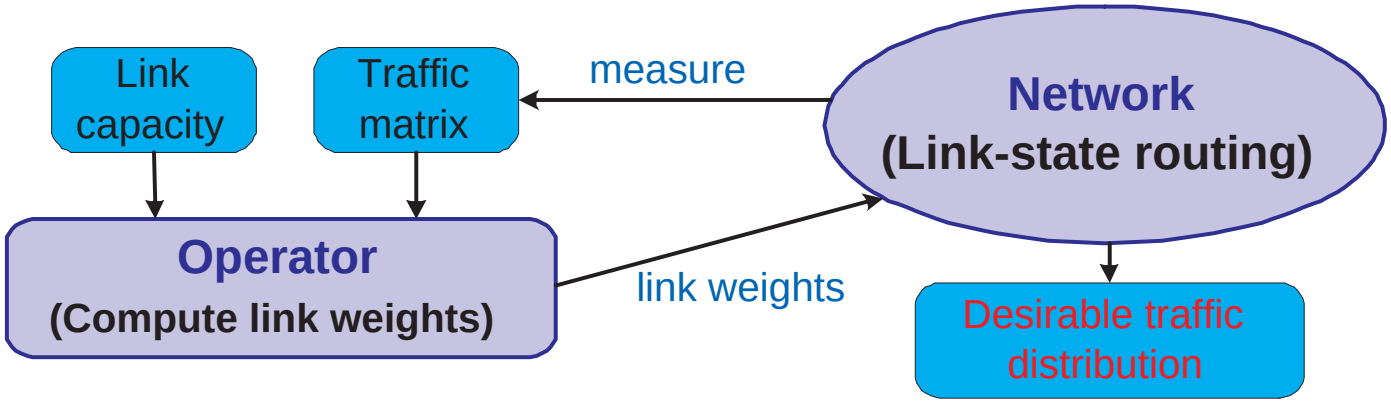
- Network operator configures link weights to guide routing
  $\Rightarrow$ Traffic Engineering

Tuning Link Weights



- Traffic Engineering (TE): based on the offered traffic matrix
  - Traffic matrix: rate of traffic between each node pair from measurement
  - Centralized and off-line
  - Network-wide optimization objective: minimizes key metrics like max link utilization

Tuning Link Weights



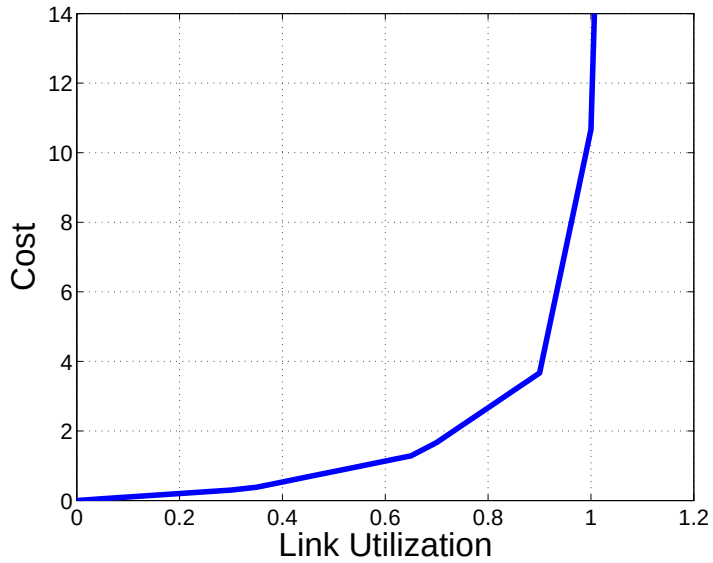
- Traffic Engineering (TE): based on the offered traffic matrix
  - Traffic matrix: rate of traffic between each node pair from measurement
  - Centralized and off-line
  - Network-wide optimization objective: minimizes key metrics like max link utilization
- Robust Traffic Engineering
  - Multiple Traffic Matrices
  - Robust against link failure/maintenance

Why Link Weights?

- Low overhead: one parameter for each unidirectional link
- Hop-by-hop forwarding: no tunneling, no history, no per-flow statistics.
- Robust: routers automatically recompute new routes in case of topology changes
- Effective: changing a few link weights is sufficient to alleviate network congestion

Objective of Traffic Engineering

- In general, Convex Objective
- Ex. 1: Minimize maximum link utilization
- Ex. 2: Minimize sum of (link) congestion cost
  - Φ congestion cost: increasing function of link utilization (typically piece-wise linear)
  - Model queuing delay



Optimal Traffic Engineering

- **Definition**: Flow-level (tunnel) of routing to minimize convex network-wide objective
- Realization with Multi-Protocol Label Switching (MPLS)
  Arbitrarily split traffic over many tunnels (not based on link weights)
- Several approaches with non-even-splitting across shortest paths
  [Wang-Wang-Zhang-01, Sridharan-Guérin-Diot-05]
  Routers cannot independently compute flow-splitting ratios from link weights.

Optimal Traffic Engineering

- **Definition**: Flow-level (tunnel) of routing to minimize convex network-wide objective
- Realization with Multi-Protocol Label Switching (MPLS)
  Arbitrarily split traffic over many tunnels (not based on link weights)
- Several approaches with non-even-splitting across shortest paths [Wang-Wang-Zhang-01, Sridharan-Guérin-Diot-05]
  Routers cannot independently compute flow-splitting ratios from link weights.
- High overhead and not self-adaptive!

Traffic Engineering with Link Weights (I)

Even-Splitting on outgoing links on shortest path (OSPF)

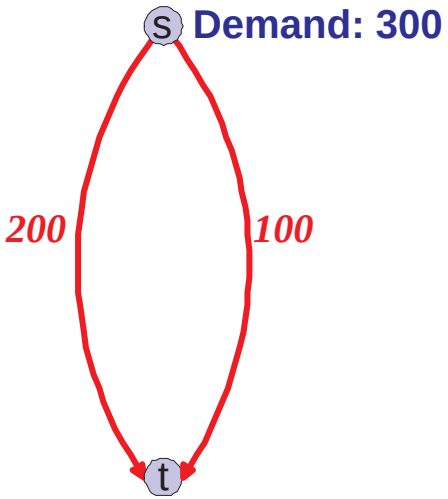
- Finding best link weights is NP-Hard! [Fortz-Thorup-00]
  - No constant-ratio approximation
  - Heuristics (Local search, etc.)
    [Fortz-Thorup-00, Ericsson-Resende-Pardalos-02]
  - Harder or even impossible for robust cases

Traffic Engineering with Link Weights (I)

Even-Splitting on outgoing links on shortest path (OSPF)

- Finding best link weights is NP-Hard! [Fortz-Thorup-00]
  - No constant-ratio approximation
  - Heuristics (Local search, etc.)
    [Fortz-Thorup-00, Ericsson-Resende-Pardalos-02]
  - Harder or even impossible for robust cases

- Best OSPF could fail to achieve optimal TE.



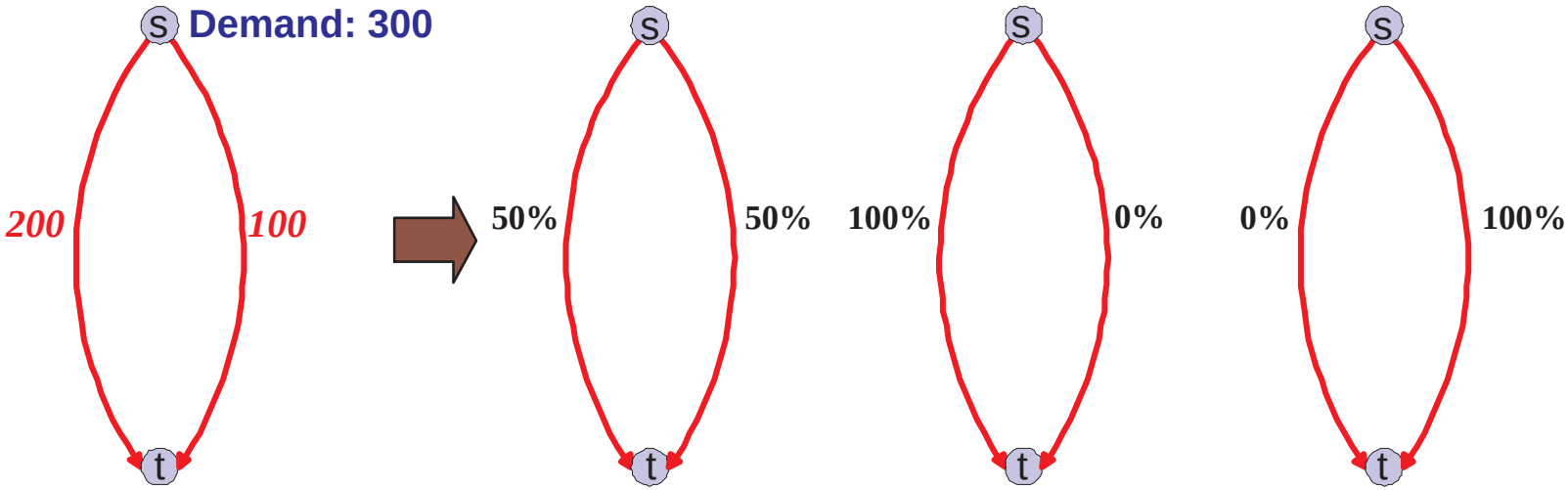
(a) Link Capacity

Traffic Engineering with Link Weights (I)

Even-Splitting on outgoing links on shortest path (OSPF)

- Finding best link weights is NP-Hard! [Fortz-Thorup-00]
  - No constant-ratio approximation
  - Heuristics (Local search, etc.)
    [Fortz-Thorup-00, Ericsson-Resende-Pardalos-02]
  - Harder or even impossible for robust cases

- Best OSPF could fail to achieve optimal TE.



(a) Link Capacity

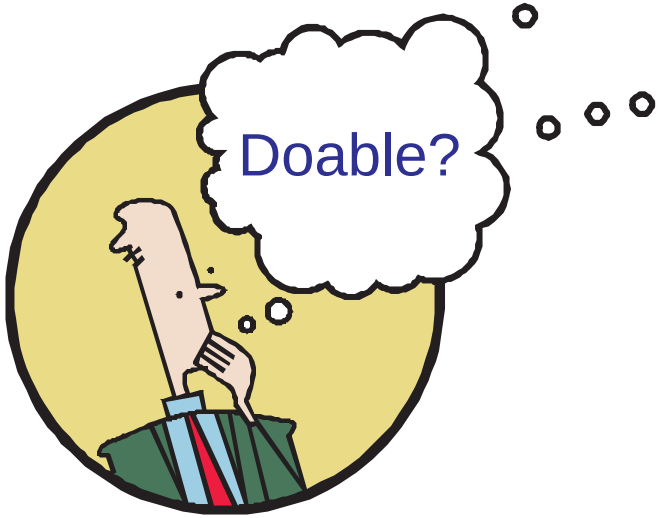
(b) Only three possible flow splitting in OSPF

Traffic Engineering with Link Weights (II)

- Link-state routing $\neq$ shortest path routing
- Non-shortest path routing
  - Exponential penalty on longer path [Fong-Gilbert-Kannan-Strauss-05]
  - Flexible flow-splitting rules simplify link-weight setting for specific networks

Trade-off between Optimality and Simplicity

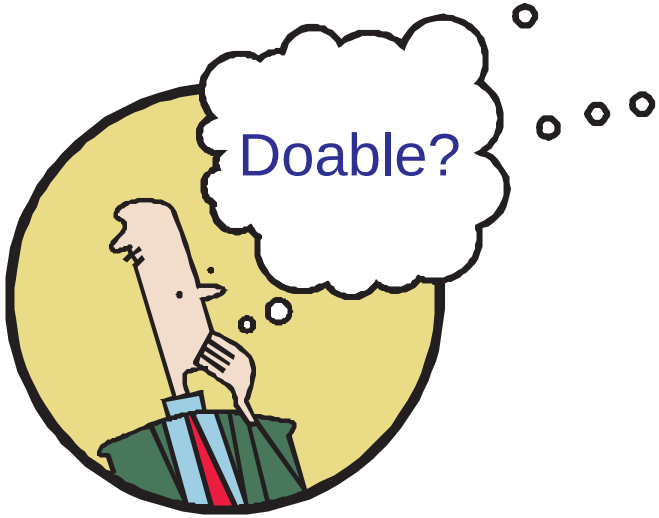
Open Questions



Optimal TE with ONLY link weights?

Find link weights in a tractable way?

Open Questions



Optimal TE with ONLY link weights?

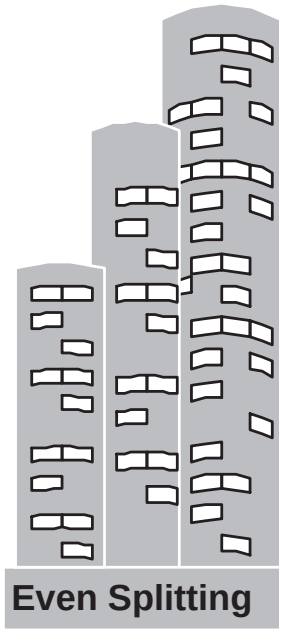
Find link weights in a tractable way?

YES !

Main Results

- **Theory**: New optimization framework, Network Entropy Maximization (NEM)
- **Application**: Our proposed link-state TE
  - Increase efficiency of using Internet capacity by 15% for a real backbone network (Abilene)
  - 2000 times faster than local search OSPF in finding the best link weights

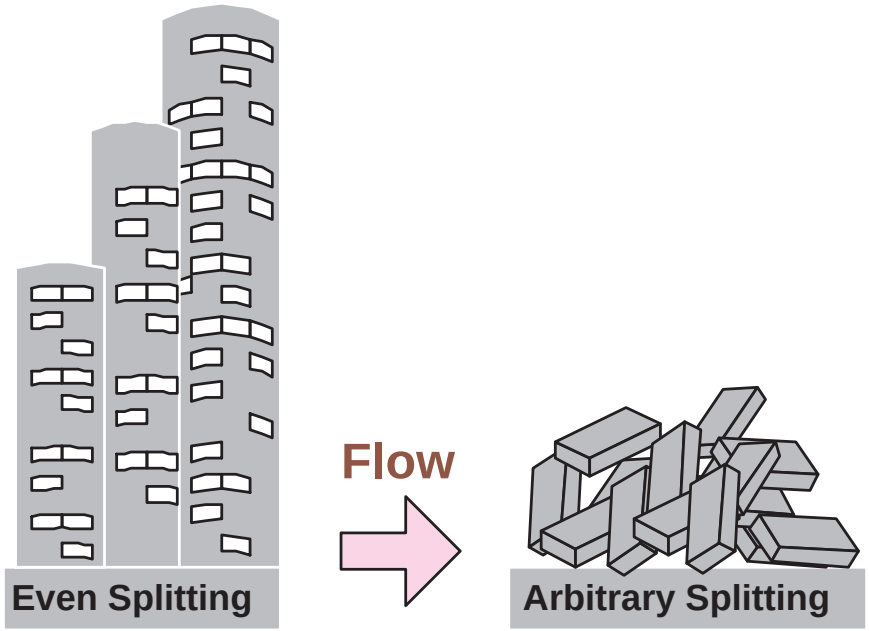
Computational Complexity of Traffic Engineering



NP-Hard

Scalable

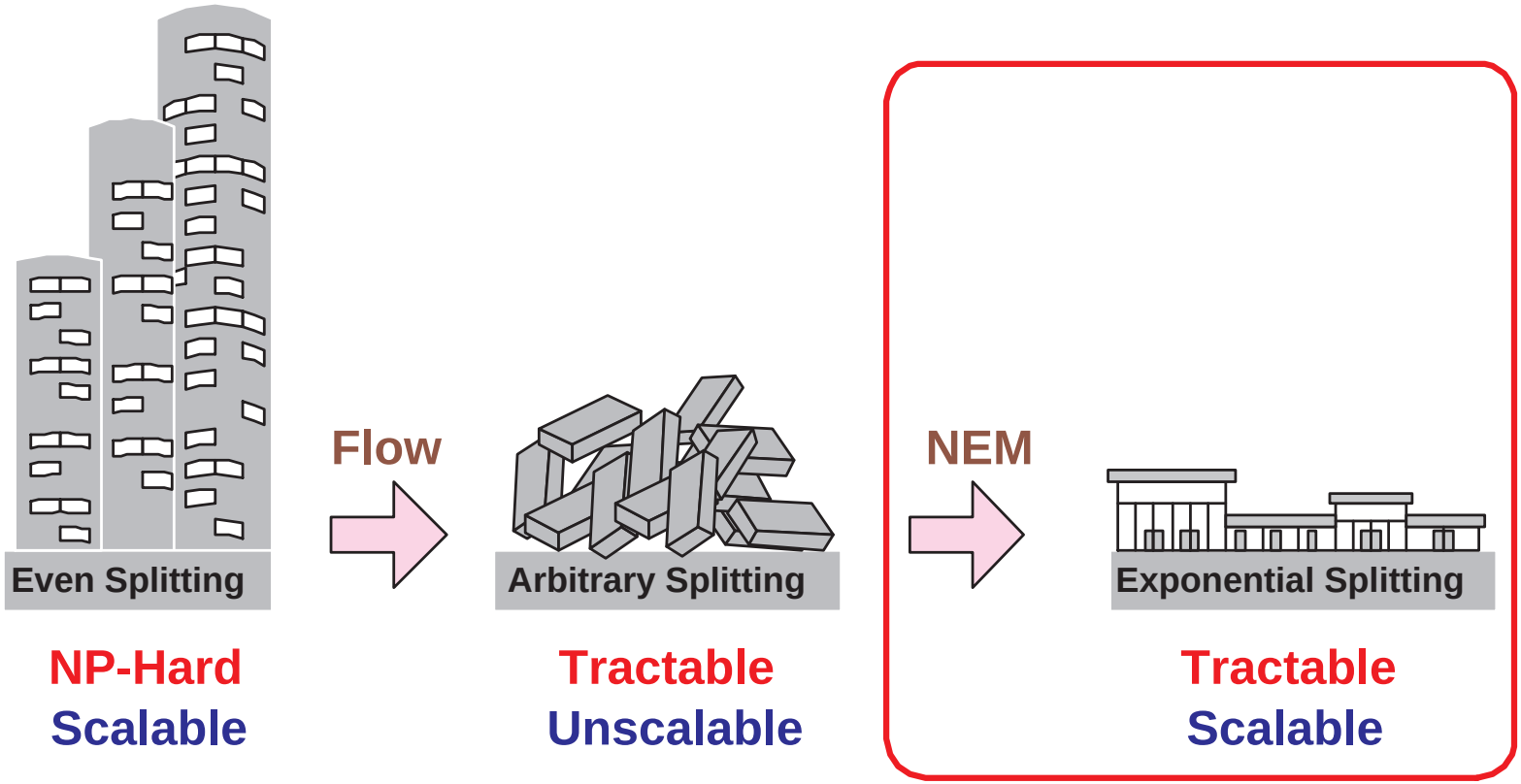
Computational Complexity of Traffic Engineering



NP-Hard
Scalable

Tractable
Unscalable

Computational Complexity of Traffic Engineering



Outline

Notation

- Directed graph: N nodes and E links
- Inputs

$D(s,t)$	Traffic demand from s to t
$c_{u,v}$	Capacity of link (u,v)

- Variables

$w_{u,v}$	Weight for link (u,v)
$f^t_{u,v}$	Commodity flow on link (u,v) destined to t
$f_{u,v}$	$\triangleq \sum_t f^t_{u,v}$, Total flow on link (u,v)

Optimal TE Via Multi-Commodity-Flow

COMMODITY Problem:

minimize	$\Phi(\{f_{u,v}, c_{u,v}\})$	convex objective
subject to	$\sum_{v:(s,v)\in\mathbb{E}} f^t_{s,v} - \sum_{u:(u,s)\in\mathbb{E}} f^t_{u,s} = D(s,t)$	flow conservation
	$f_{u,v} \triangleq \sum_{t\in\mathbb{V}} f^t_{u,v} \leq c_{u,v}$	capacity constraint
variables	$f_{u,v} \geq f^t_{u,v} \geq 0.$	link flow, commodity flow
input	$D(s,t), c_{u,v}$	demand, capacity

Optimal TE Via Multi-Commodity-Flow

COMMODITY Problem:

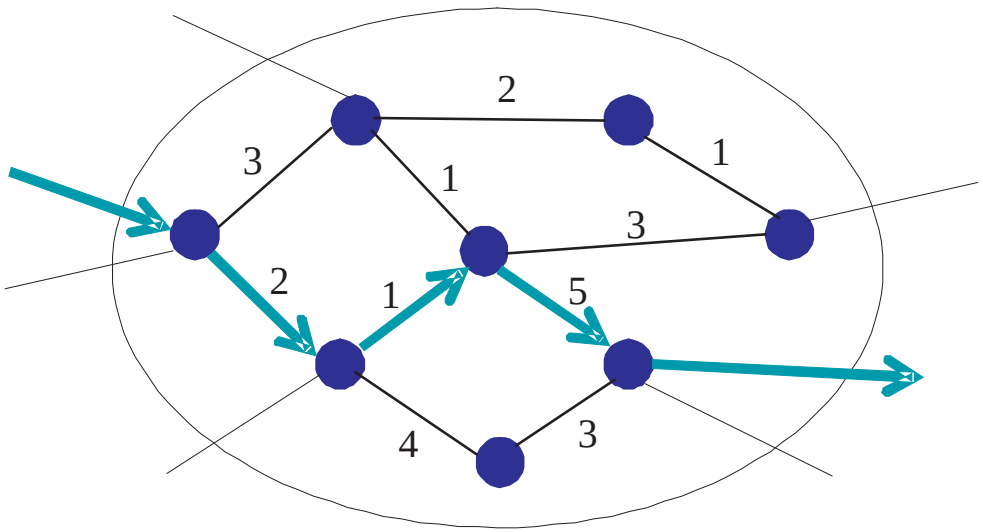
minimize	$\Phi(\{f_{u,v}, c_{u,v}\})$	convex objective
subject to	$\sum_{v:(s,v)\in\mathbb{E}} f^t_{s,v} - \sum_{u:(u,s)\in\mathbb{E}} f^t_{u,s} = D(s,t)$	flow conservation
	$f_{u,v} \triangleq \sum_{t\in\mathbb{V}} f^t_{u,v} \leq c_{u,v}$	capacity constraint
variables	$f_{u,v} \geq f^t_{u,v} \geq 0.$	link flow, commodity flow
input	$D(s,t), c_{u,v}$	demand, capacity

- Convex optimization (efficiently solvable).
- Can be realized with explicit routing: set up N^2E tunnels
- Link-state routing: E parameters?

Forward Packets Based on Link Weights

Achievable information for router u

$w_{u,v}$	Weights for all links
d_u^t	The shortest distance from u to t. $d_t^t = 0$
$h_{u,v}^t$	Shortest distance gap, $h_{u,v}^t = d_v^t + w_{u,v} - d_u^t$

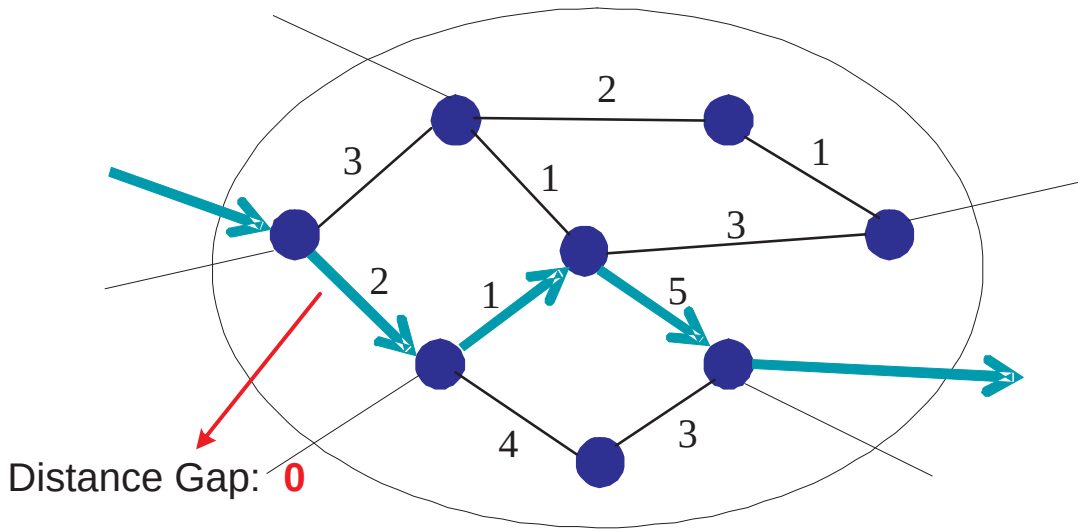


Path length= 8

Forward Packets Based on Link Weights

Achievable information for router u

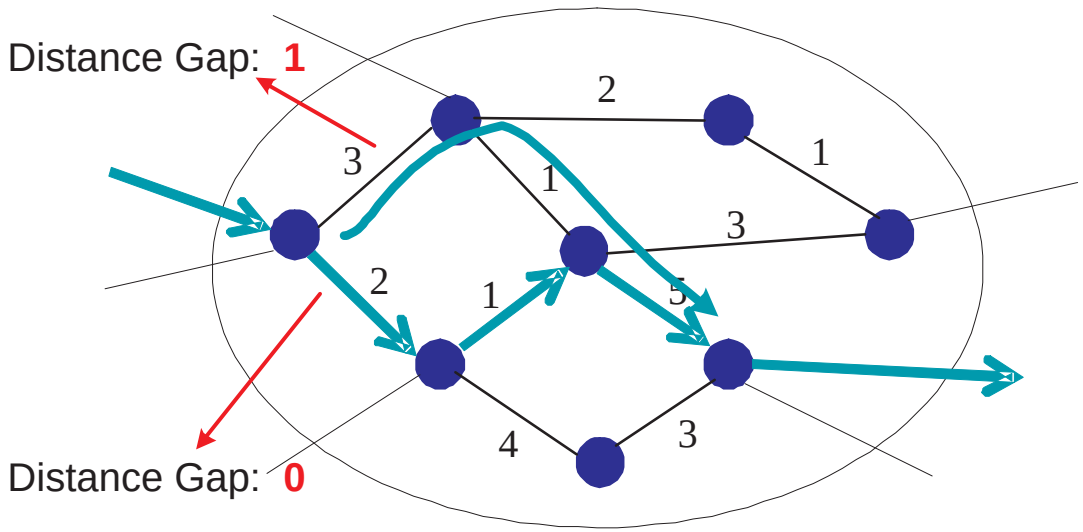
$w_{u,v}$	Weights for all links
d_u^t	The shortest distance from u to t. $d_t^t = 0$
$h_{u,v}^t$	Shortest distance gap, $h_{u,v}^t = d_v^t + w_{u,v} - d_u^t$



Forward Packets Based on Link Weights

Achievable information for router u

$w_{u,v}$	Weights for all links
d_u^t	The shortest distance from u to t. $d_t^t = 0$
$h_{u,v}^t$	Shortest distance gap, $h_{u,v}^t = d_v^t + w_{u,v} - d_u^t$



Path length= 8

Traffic-Splitting Function

- Traffic-splitting function ($\Gamma(h^t_{u,v})$): relative amount of flow distributed across outgoing links

Traffic-Splitting Function

- Traffic-splitting function ($\Gamma(h_{u,v}^t)$): relative amount of flow distributed across outgoing links
- f_u^t: total flow (destined to t) at node u (incoming + self-originated)
- The outgoing flow (destined to t) on link (u, v)

$$f_{u,v}^t = f_u^t \frac{\Gamma(h_{u,v}^t)}{\sum_{(u,j)\in\mathbb{E}} \Gamma(h_{u,j}^t)}$$

Traffic-Splitting Function with Even-splitting

- Even splitting among links on shortest paths (e.g., OSPF)

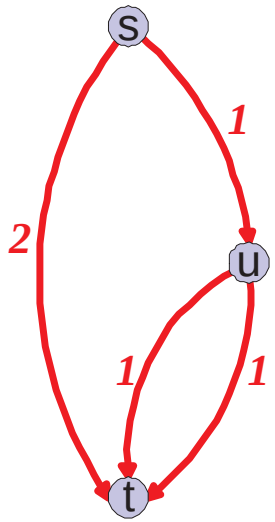
$$\Gamma_O(h^t_{u,v}) = \begin{cases} 1 & \text{if } h^t_{u,v} = 0, \\ 0 & \text{if } h^t_{u,v} > 0. \end{cases}$$

Link and Path-Based Flow Splitting

- Link-State Routing: split traffic destined to t
  - Link-based: across all outgoing links of s
    - ⋆ Single-Destination: [Fong-Gilbert-Kannan-Strauss-05]
    - ⋆ Multi-Destination: DEFT [Xu-Chiang-Rexford-Infocom-07]
  - Path-based: among all possible paths from s to t
    PEFT [Xu-Chiang-Rexford-Infocom-08]

Link and Path-Based Flow Splitting

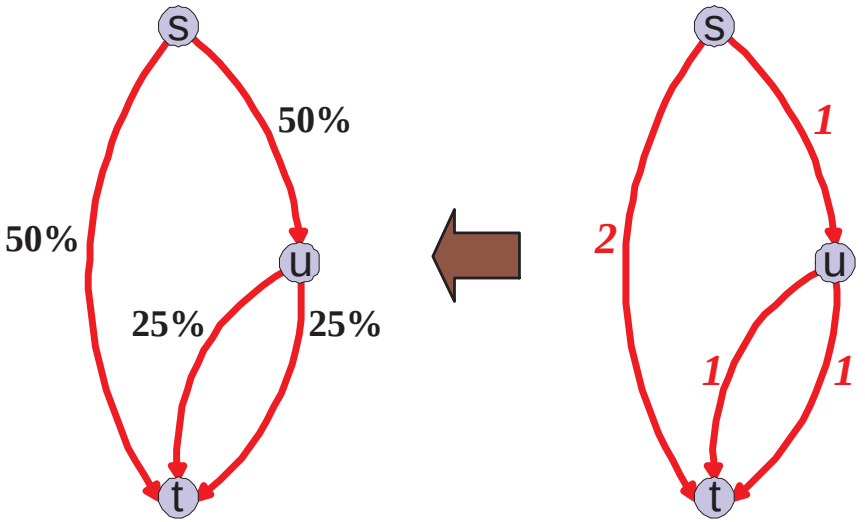
- Link-State Routing: split traffic destined to t
  - Link-based: across all outgoing links of s
    - ★ Single-Destination: [Fong-Gilbert-Kannan-Strauss-05]
    - ★ Multi-Destination: DEFT [Xu-Chiang-Rexford-Infocom-07]
  - Path-based: among all possible paths from s to t
    PEFT [Xu-Chiang-Rexford-Infocom-08]



(a) Link Weights

Link and Path-Based Flow Splitting

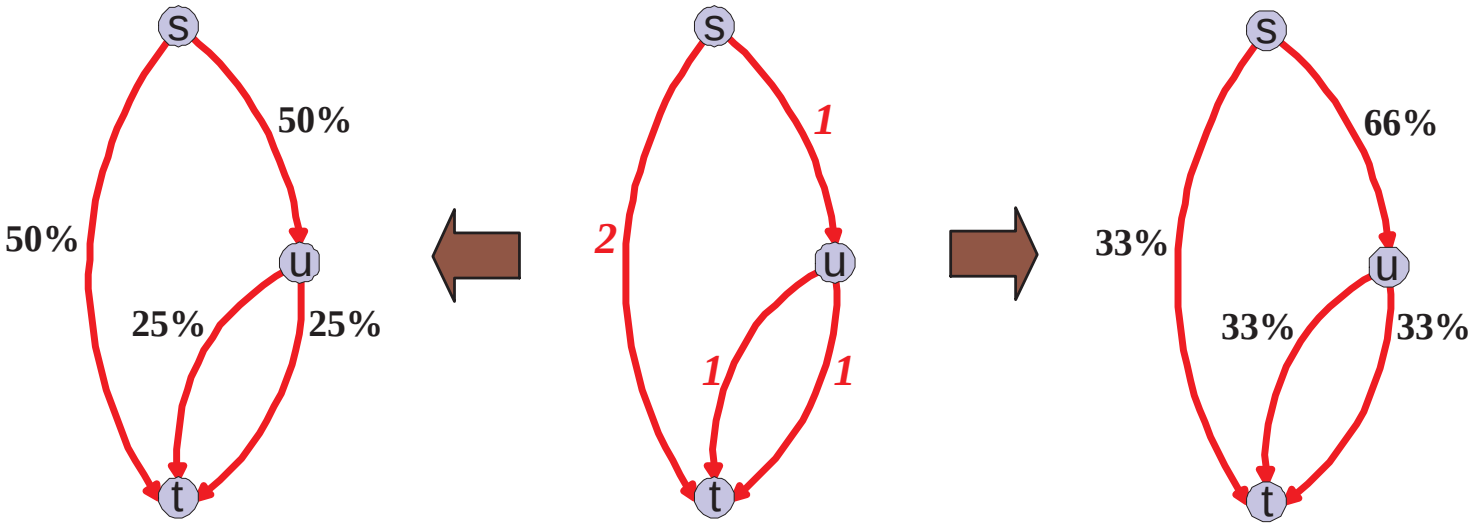
- Link-State Routing: split traffic destined to t
  - Link-based: across all outgoing links of s
    - ★ Single-Destination: [Fong-Gilbert-Kannan-Strauss-05]
    - ★ Multi-Destination: DEFT [Xu-Chiang-Rexford-Infocom-07]
  - Path-based: among all possible paths from s to t
    PEFT [Xu-Chiang-Rexford-Infocom-08]



(b) Link-Based Splitting (a) Link Weights

Link and Path-Based Flow Splitting

- Link-State Routing: split traffic destined to t
  - Link-based: across all outgoing links of s
    - ★ Single-Destination: [Fong-Gilbert-Kannan-Strauss-05]
    - ★ Multi-Destination: DEFT [Xu-Chiang-Rexford-Infocom-07]
  - Path-based: among all possible paths from s to t
    PEFT [Xu-Chiang-Rexford-Infocom-08]



(b) Link-Based Splitting (a) Link Weights (c) Path-based Splitting

Path-based Exponentially-weighted Flow Splitting (PEFT)

- **PEFT**: probability of using i-th path $\propto e^{-p_i}$, p_i (path length)
- Corresponding traffic splitting function without path enumeration?

Path-based Exponentially-weighted Flow Splitting (PEFT)

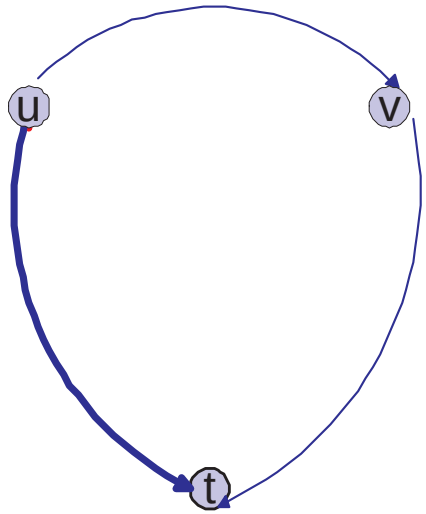
- **PEFT**: probability of using i-th path $\propto e^{-p_i}$, p_i (path length)
- Corresponding traffic splitting function without path enumeration?
- Commodity Quota (Υ_u^t):
  - Equivalent number of shortest paths from u to t, $\Upsilon_t^t \triangleq 1$
  - In PEFT, treat a non-shortest path (p) as $e^{-(p-d_u^t)}$ shortest path (d_u^t)

Path-based Exponentially-weighted Flow Splitting (PEFT)

- **PEFT**: probability of using i-th path $\propto e^{-p_i}$, p_i (path length)
- Corresponding traffic splitting function without path enumeration?
- Commodity Quota (Υ_u^t):
  - Equivalent number of shortest paths from u to t, $\Upsilon_t^t \triangleq 1$
  - In PEFT, treat a non-shortest path (p) as $e^{-(p-d_u^t)}$ shortest path (d_u^t)
- $$\begin{aligned}\Upsilon_u^t &\triangleq \sum_i e^{-(p_{u,t}^i - d_u^t)} \\ &= \sum_i \left(\sum_{(u,v)\in\mathbb{E}} e^{-(p_{u,t}^i - w_{u,v} - d_v^t + d_v^t + w_{u,v} - d_u^t)} \right) \\ &= \sum_{(u,v)\in\mathbb{E}} \left(e^{-h_{u,v}^t} \Upsilon_v^t \right)\end{aligned}$$

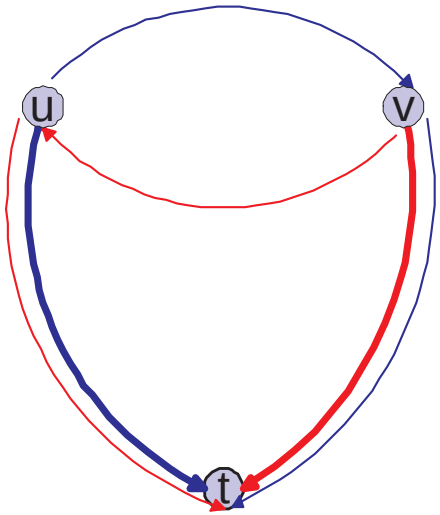
Exact and Downward PEFT

- Exact PEFT: $\Gamma_{PX}(h^t_{u,v}) = \Upsilon^t_v e^{-h^t_{u,v}}$
- Traffic may move backwards $\Rightarrow$ loopback flow in $\Gamma_{PX}(\cdot)$



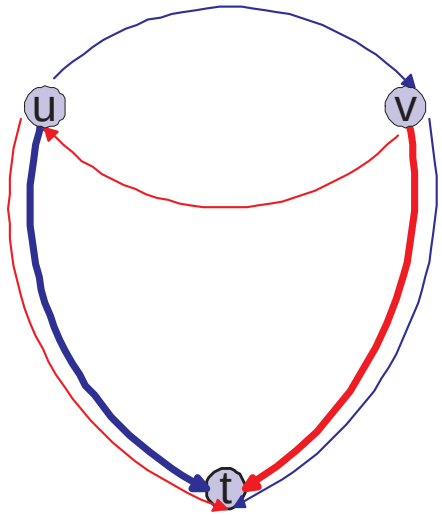
Exact and Downward PEFT

- Exact PEFT: $\Gamma_{PX}(h^t_{u,v}) = \Upsilon^t_v e^{-h^t_{u,v}}$
- Traffic may move backwards $\Rightarrow$ loopback flow in $\Gamma_{PX}(\cdot)$



Exact and Downward PEFT

- Exact PEFT: $\Gamma_{PX}(h^t_{u,v}) = \Upsilon^t_v e^{-h^t_{u,v}}$
- Traffic may move backwards $\Rightarrow$ loopback flow in $\Gamma_{PX}(\cdot)$



- Prevent loopback flow, only directs traffic on "downward" paths
- Downward PEFT

$$\Gamma_{PD}(h^t_{u,v}) = \begin{cases} \Upsilon^t_v e^{-h^t_{u,v}} & \text{if } d^t_u > d^t_v, \\ 0 & \text{otherwise.} \end{cases}$$

Link-State Routing Can Achieve Optimal TE

Theorem (Optimal TE is achievable)

Optimal TE for a given traffic matrix can be realized with link weights, by using traffic-splitting function $\Gamma_{PX}(\cdot)$ in link-state routing protocols.

- [Xu-Chiang-Rexford-Infocom-08]
- Algorithm first, then proof.

Necessary Capacity

- Necessary Capacity
  - Solve COMMODITY problem to compute optimal traffic distribution for a given traffic matrix
  - $\widetilde{c}_{u,v} \triangleq f_{u,v}$: Minimal set of link capacities to realize optimal TE

Necessary Capacity

- Necessary Capacity
  - Solve COMMODITY problem to compute optimal traffic distribution for a given traffic matrix
  - $\widetilde{c}_{u,v} \triangleq f_{u,v}$: Minimal set of link capacities to realize optimal TE
- Set link weights with only necessary capacities

Algorithm for Optimizing Link Weights

Optimize Over Link Weights

1: Compute necessary capacities $\widetilde{\mathbf{c}}$ by solving COMMODITY problem
2: $\mathbf{w} \leftarrow$ Any set of link weights
3: $\mathbf{f} \leftarrow$ Traffic_Distribution($\mathbf{w}$)
4: **while** $\mathbf{f} \neq \widetilde{\mathbf{c}}$ **do**
5: $\quad \mathbf{w} \leftarrow$ Link_Weight_Update($\mathbf{f}$)
6: $\quad \mathbf{f} \leftarrow$ Traffic_Distribution($\mathbf{w}$)
7: **end while**

Algorithm for Optimizing Link Weights

Optimize Over Link Weights

1: Compute necessary capacities $\widetilde{\mathbf{c}}$ by solving COMMODITY problem
2: $\mathbf{w} \leftarrow$ Any set of link weights
3: $\mathbf{f} \leftarrow$ Traffic_Distribution($\mathbf{w}$)
4: **while** $\mathbf{f} \neq \widetilde{\mathbf{c}}$ **do**
5: $\quad \mathbf{w} \leftarrow$ Link_Weight_Update($\mathbf{f}$)
6: $\quad \mathbf{f} \leftarrow$ Traffic_Distribution($\mathbf{w}$)
7: **end while**

Link-Weight_Update(f)

1: **for each** link (u, v) **do**
2: $\quad w_{u,v} \leftarrow w_{u,v} - \alpha\left(\widetilde{c}_{u,v} - f_{u,v}\right)$
3: **end for**
4: Return new link weights $\mathbf{w}$

Traffic Distribution with Exact Exponential Splitting

- Construct all-pairs shortest paths (e.g. with Floyd-Warshall algorithm) and compute $\Gamma_{PX}(h^t_{u,v})$.
- For each destination t, at node u
  outgoing flow = incoming flow + self-originated flow
- For each t, solve N equations with N variables.
  Time complexity $O(N^4)$

Traffic Distribution with Exact Exponential Splitting

- Construct all-pairs shortest paths (e.g. with Floyd-Warshall algorithm) and compute $\Gamma_{PX}(h^t_{u,v})$.
- For each destination t, at node u
  outgoing flow = incoming flow + self-originated flow
- For each t, solve N equations with N variables.
  Time complexity $O(N^4)$
- **Link weights realizing Optimal TE can be found tractably!**

Traffic Distribution with Downward Exponential Splitting

- Optimal traffic distribution has no loopback flow, i.e. negligible in last iteration
- Preventing loopback flow in intermediate iterations could speed up convergence
- Replace $\Gamma_{PX}(\cdot)$ with $\Gamma_{PD}(\cdot)$: forward traffic only on "downward" paths

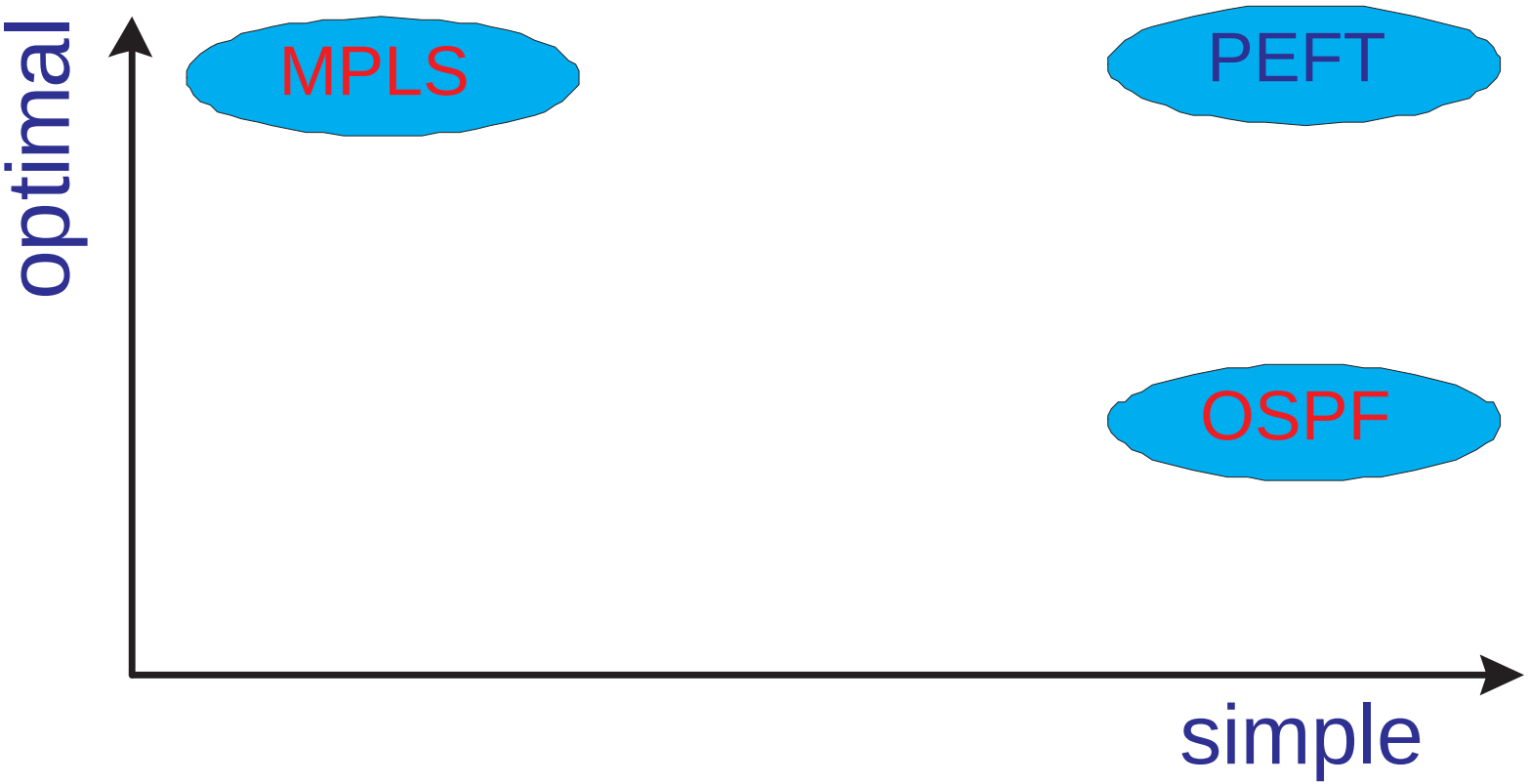
Traffic Distribution with Downward Exponential Splitting

- Optimal traffic distribution has no loopback flow, i.e. negligible in last iteration
- Preventing loopback flow in intermediate iterations could speed up convergence
- Replace $\Gamma_{PX}(\cdot)$ with $\Gamma_{PD}(\cdot)$: forward traffic only on "downward" paths
- For each destination t, remove link (u, v) where $d_u^t \leq d_v^t$, then acyclic network
- Summarize total flow and split it across outgoing links for each node with the decreasing topological order.
  Time complexity $O(N(N + E))$

Traffic Distribution with Downward Exponential Splitting

- Optimal traffic distribution has no loopback flow, i.e. negligible in last iteration
- Preventing loopback flow in intermediate iterations could speed up convergence
- Replace $\Gamma_{PX}(\cdot)$ with $\Gamma_{PD}(\cdot)$: forward traffic only on "downward" paths
- For each destination t, remove link (u, v) where $d_u^t \leq d_v^t$, then acyclic network
- Summarize total flow and split it across outgoing links for each node with the decreasing topological order.
  Time complexity $O(N(N+E)) \ll O(N^4)$ with $\Gamma_{PX}(\cdot)$
- **Practical algorithm even for very large networks**

Achieve Both Optimality and Simplicity

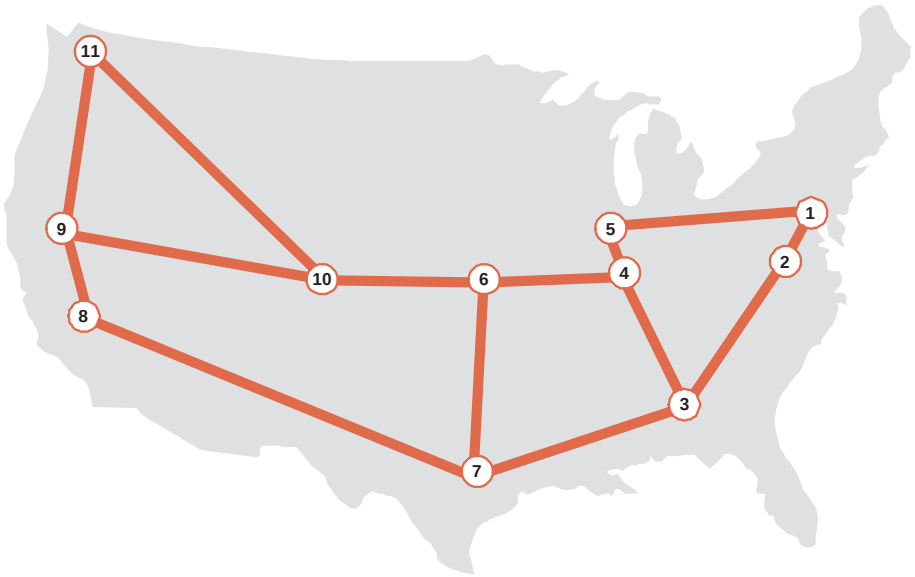


Outline

Traffic Engineering Schemes

- Optimal TE: Solve COMMODITY problem as a Linear Program (Tunnel-based)
- PEFT TE: Our algorithm (Link-weight-based)
- OSPF TE: Local search [Fortz-Thorup-2000] (Link-weight-based)

Network Topologies



Abilene Network

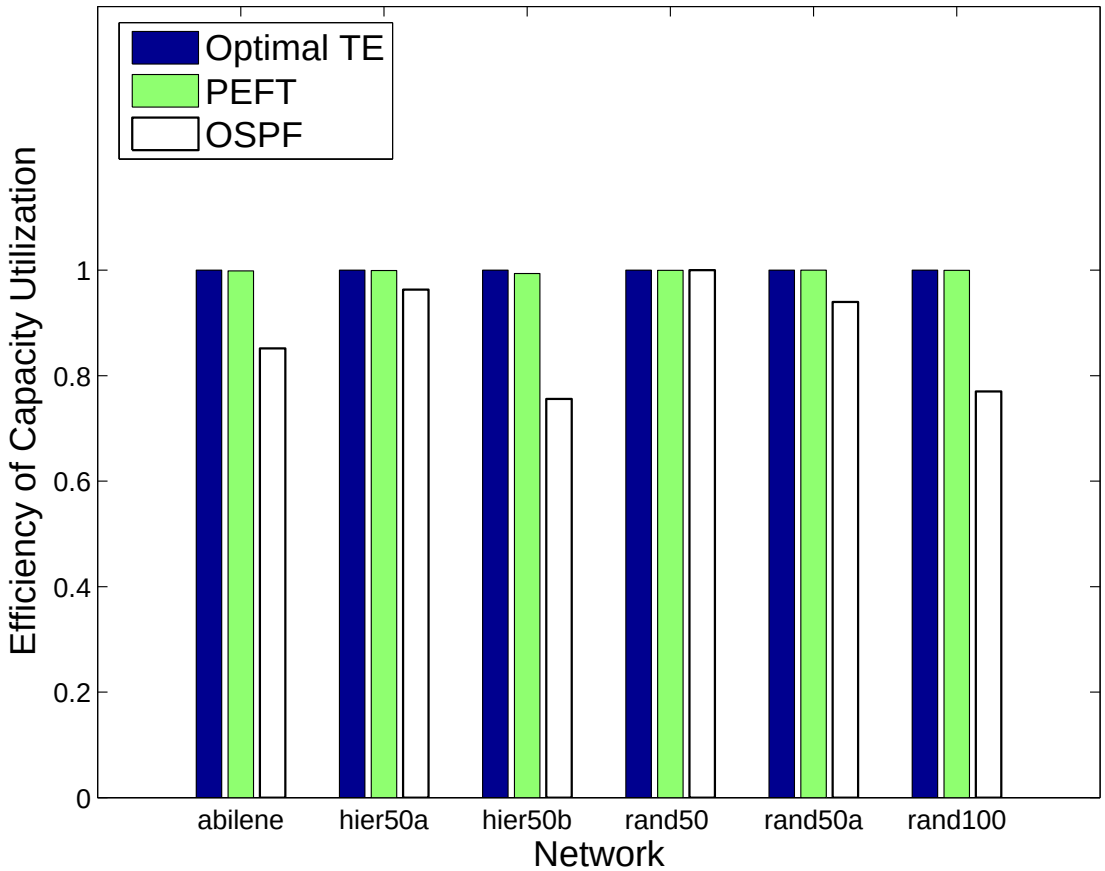
Name	Topology	Node #	Link #	Link Capacity
abilene	Backbone	11	28	10Gbps
hier50a	2-level	50	148	local access(200), long-haul (1000)
hier50b	2-level	50	212	local access(200), long-haul (1000)
rand50	Random	50	228	1000
rand50a	Random	50	245	1000
rand100	Random	100	403	1000

Traffic Matrices

- Abilene Network: measured data on Nov. 15th, 2005
- Other networks: same as [Fortz-Thorup-2000]
- 7 test cases for each network: uniformly decrease link capacity/increase demand

Minimize Maximum Link Utilization

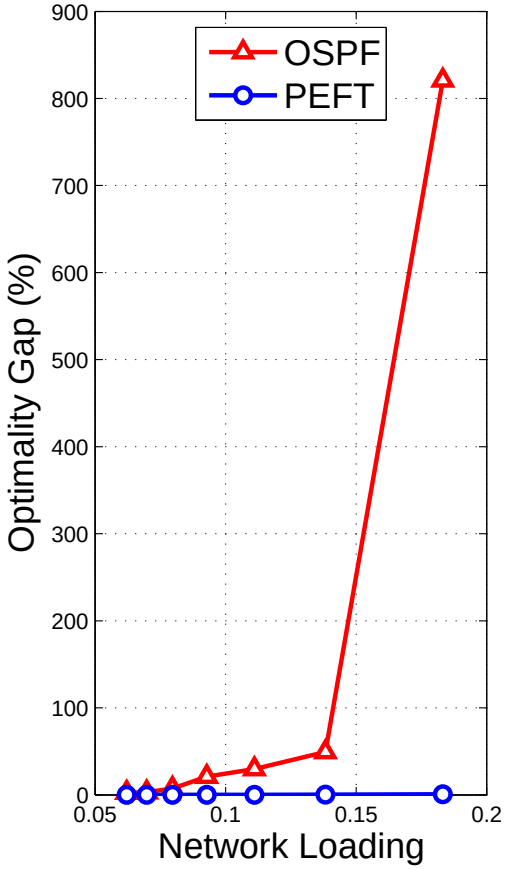
- Efficiency of capacity utilization: Percentage of traffic demand satisfied when a link utilization reaches 100%.
- PEFT achieves optimal TE, and increases Internet capacity over OSPF by 15% for Abilene and 24% for Hier50b



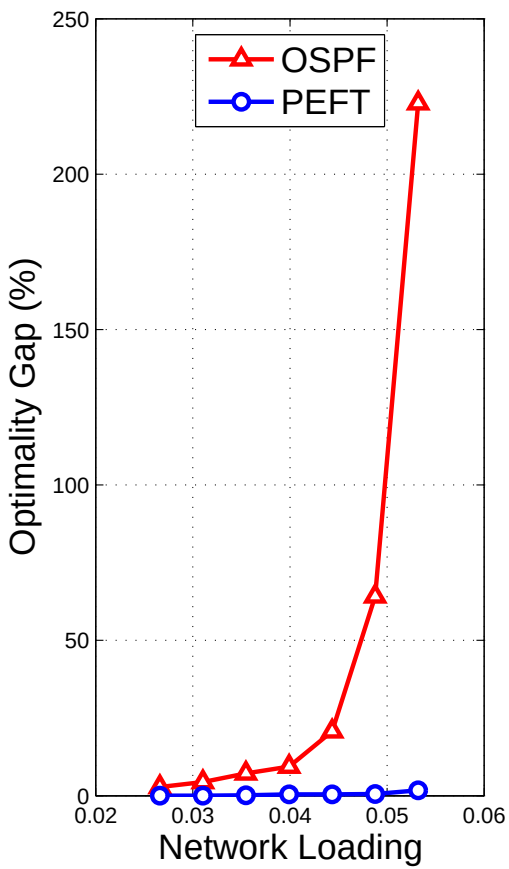
Minimize Total Link Congestion Cost

- Optimality gap (compared against optimal TE)

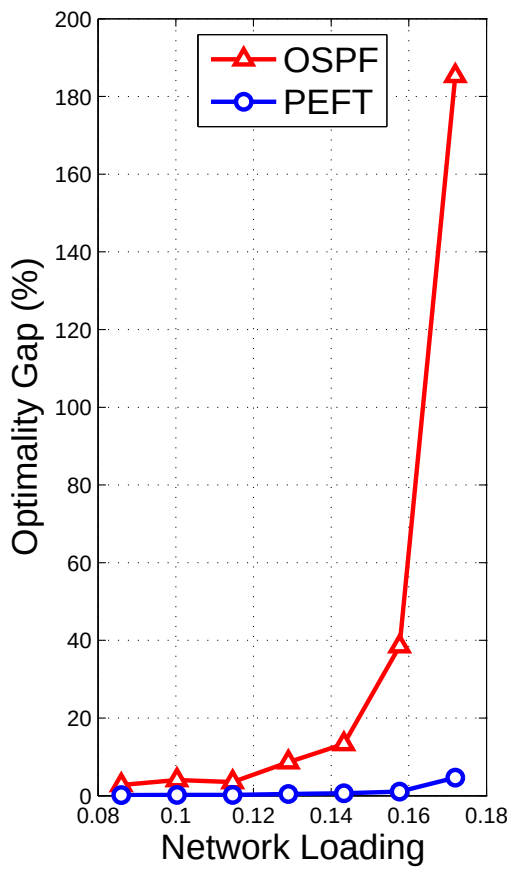
abilene



hier50b



rand100

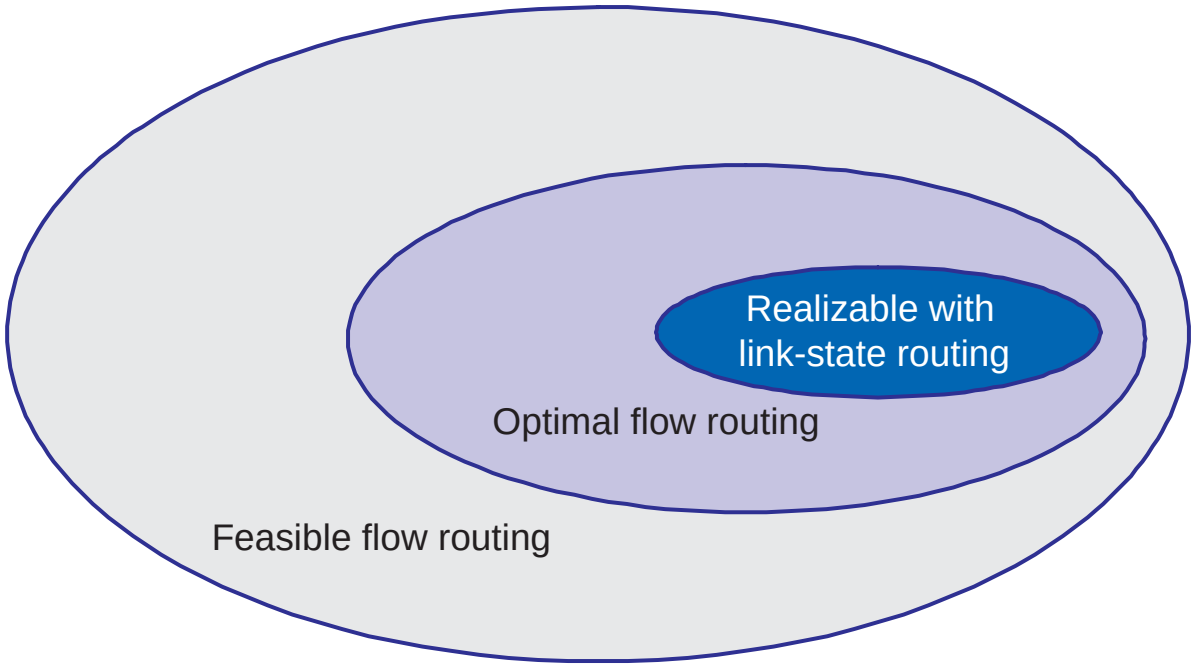


Running Time

- TE with PEFT requires at most 2 minutes even for the largest network tested.
- The algorithm to find link weights for PEFT routing is 2000 times faster than local search algorithms (public version in TOTEM) for OSPF routing.

Outline

Intuition Behind the Theory



- Numerous ways of flow-level routing to realize optimal TE (different traffic distribution on the paths)
- Choose the flow-level routing which can be realized with link-state routing.
- How? Pick an additional objective function for these optimal flow-level routings

Optimal Flow-level Routing

- Optimal TE
  - Solve COMMODITY problem
  - Only keep Necessary Capacity: $\widetilde{c}_{u,v} \triangleq f_{u,v}$,
    - ★ Any feasible flow-level routing is optimal in minimizing $\Phi(\{f_{u,v}\})$
    - ★ Ignore $\Phi()$ objective thereafter

Optimal Flow-level Routing

- Optimal TE
  - Solve COMMODITY problem
  - Only keep Necessary Capacity: $\widetilde{c}_{u,v} \triangleq f_{u,v}$,
    - ⋆ Any feasible flow-level routing is optimal in minimizing $\Phi(\{f_{u,v}\})$
    - ⋆ Ignore $\Phi()$ objective thereafter
- New objective
  maximize network entropy

Network Entropy Maximization

- Assume we can enumerate all the paths from s to t, $P^i_{s,t}$. (only for analysis purpose)
- $x^i_{s,t}$: probability (fraction) of forwarding a packet of demand $D(s,t)$ to the i-th path ($P^i_{s,t}$)

subject to	$\sum_{s,t,i:(u,v)\in P^i_{s,t}} D(s,t)x^i_{s,t} \leq \widetilde{c}_{u,v}$	capacity constraint
	$\sum_i x^i_{s,t} = 1$	flow conservation
variables	$0 \leq x^i_{s,t} \leq 1.$	forwarding probability

Network Entropy Maximization

- Assume we can enumerate all the paths from s to t, $P_{s,t}^i$. (only for analysis purpose)
- $x_{s,t}^i$: probability (fraction) of forwarding a packet of demand $D(s,t)$ to the i-th path ($P_{s,t}^i$)
- $z(x) = -x \log x$: Entropy function

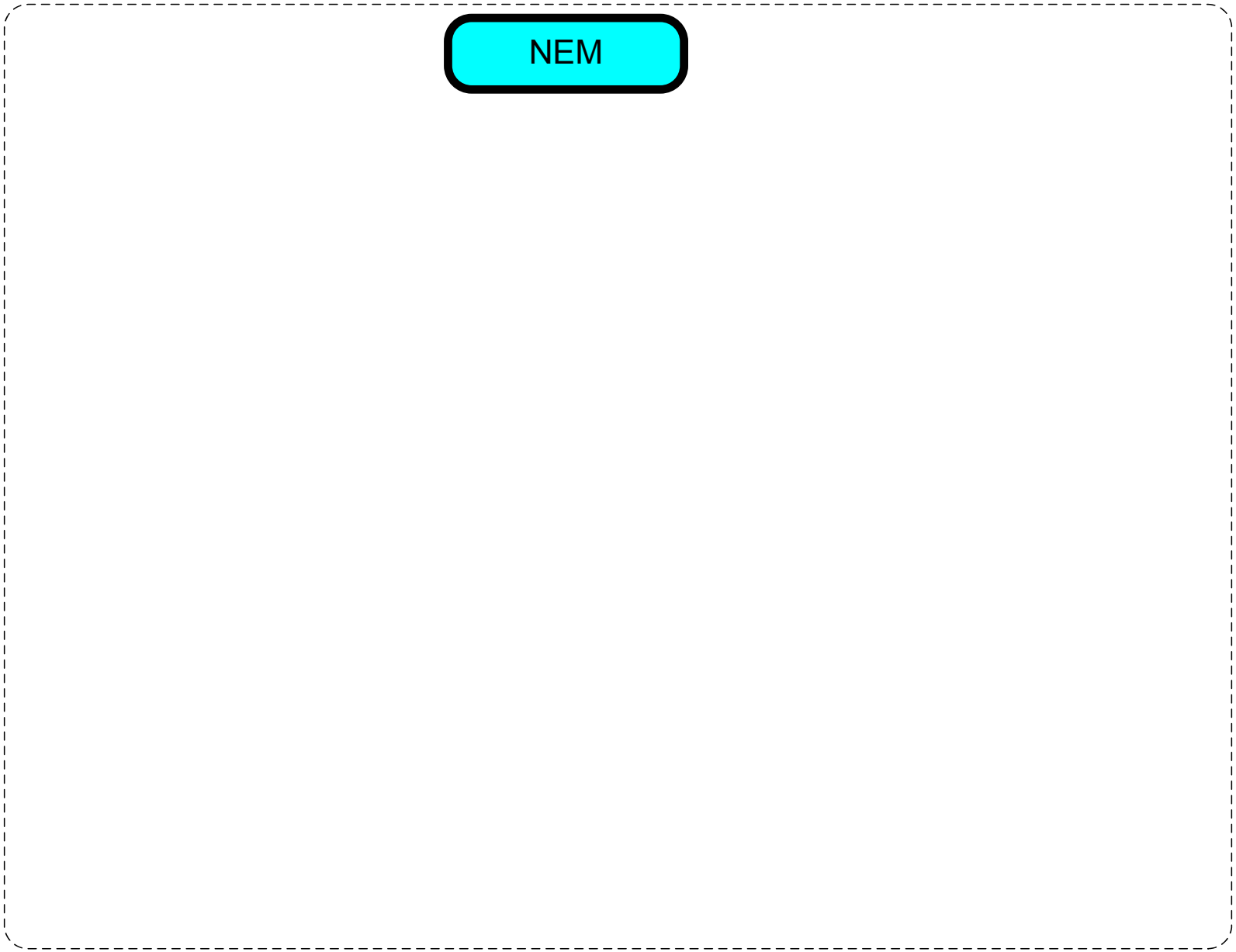
Network Entropy Maximization (NEM)

maximize	$\sum_{s,t} D(s,t) \left(\sum_{P_{s,t}^i} z(x_{s,t}^i) \right)$	total entropy
subject to	$\sum_{s,t,i:(u,v)\in P_{s,t}^i} D(s,t) x_{s,t}^i \leq \widetilde{c}_{u,v}$	capacity constraint
	$\sum_i x_{s,t}^i = 1$	flow conservation
variables	$0 \leq x_{s,t}^i \leq 1.$	forwarding probability

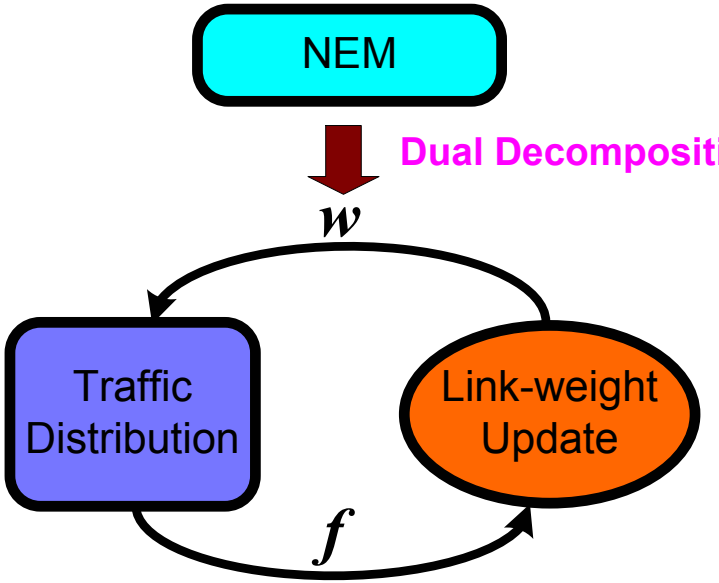
NEM features

- NEM problem always has a global optimal solution.
  - Feasible solution: any optimal solution of COMMODITY problem
  - $z(x)$ is a concave function
  - Convex Optimization
- Solving directly is not efficient (Exponential path enumeration)

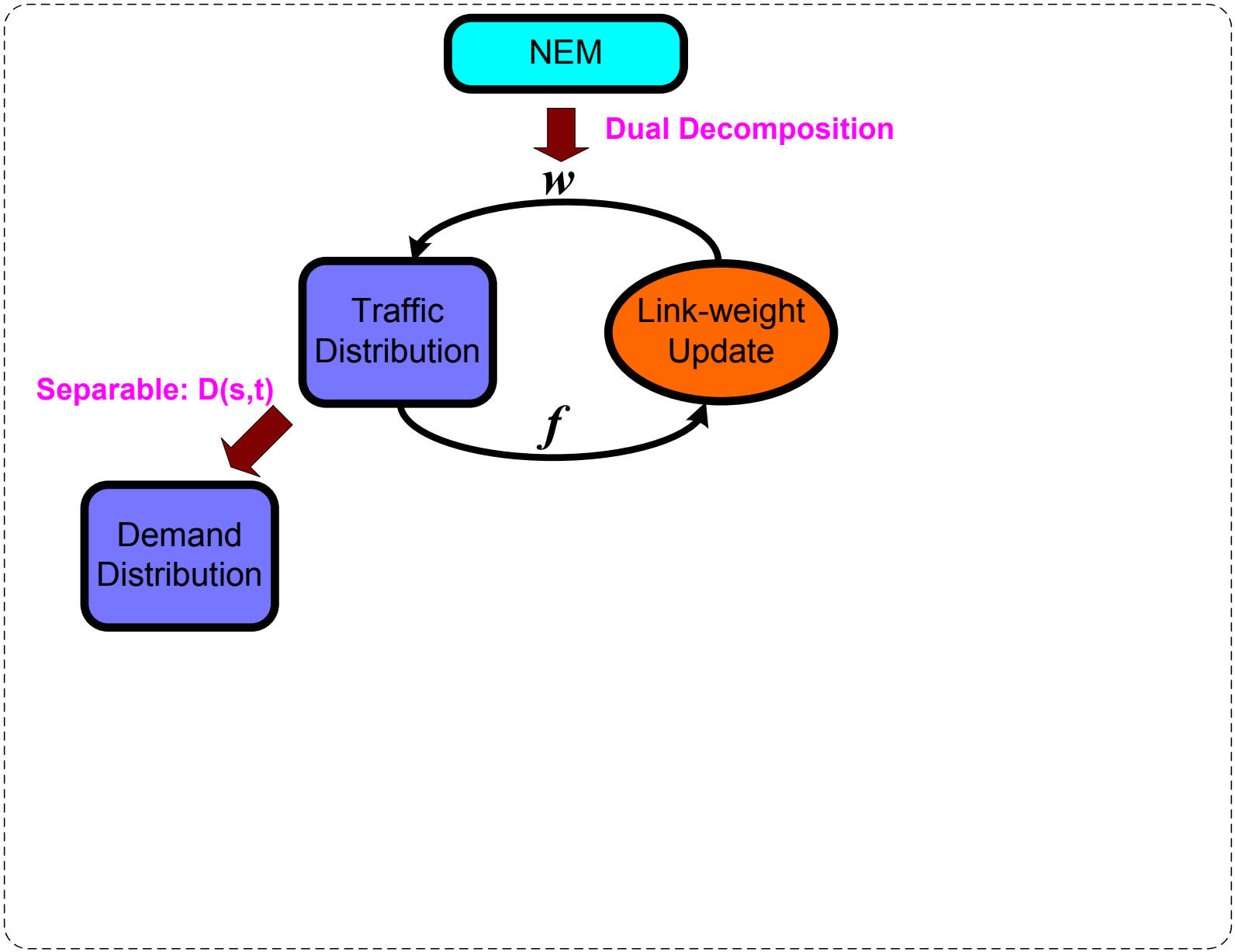
Solve NEM with Dual Decomposition



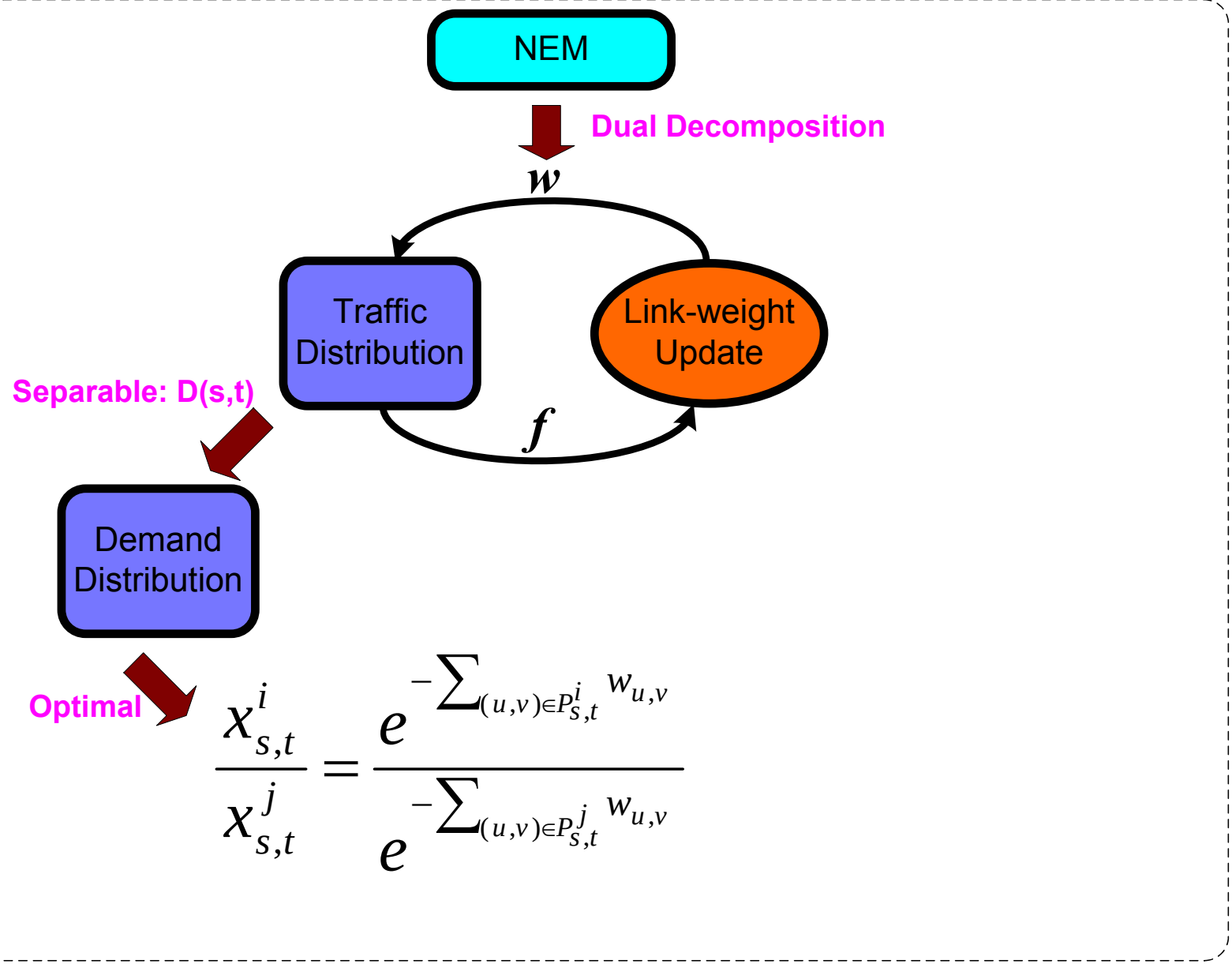
Solve NEM with Dual Decomposition



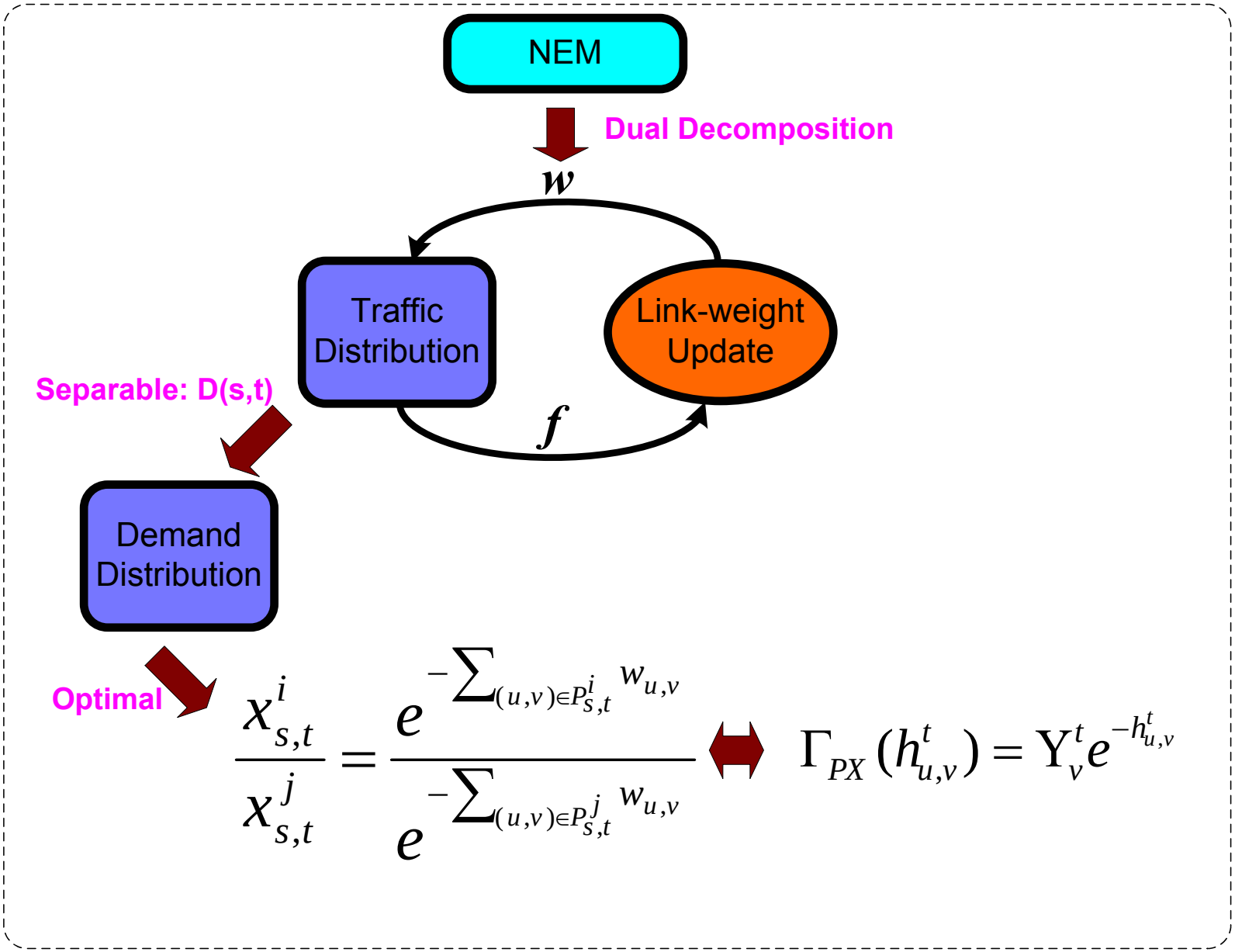
Solve NEM with Dual Decomposition



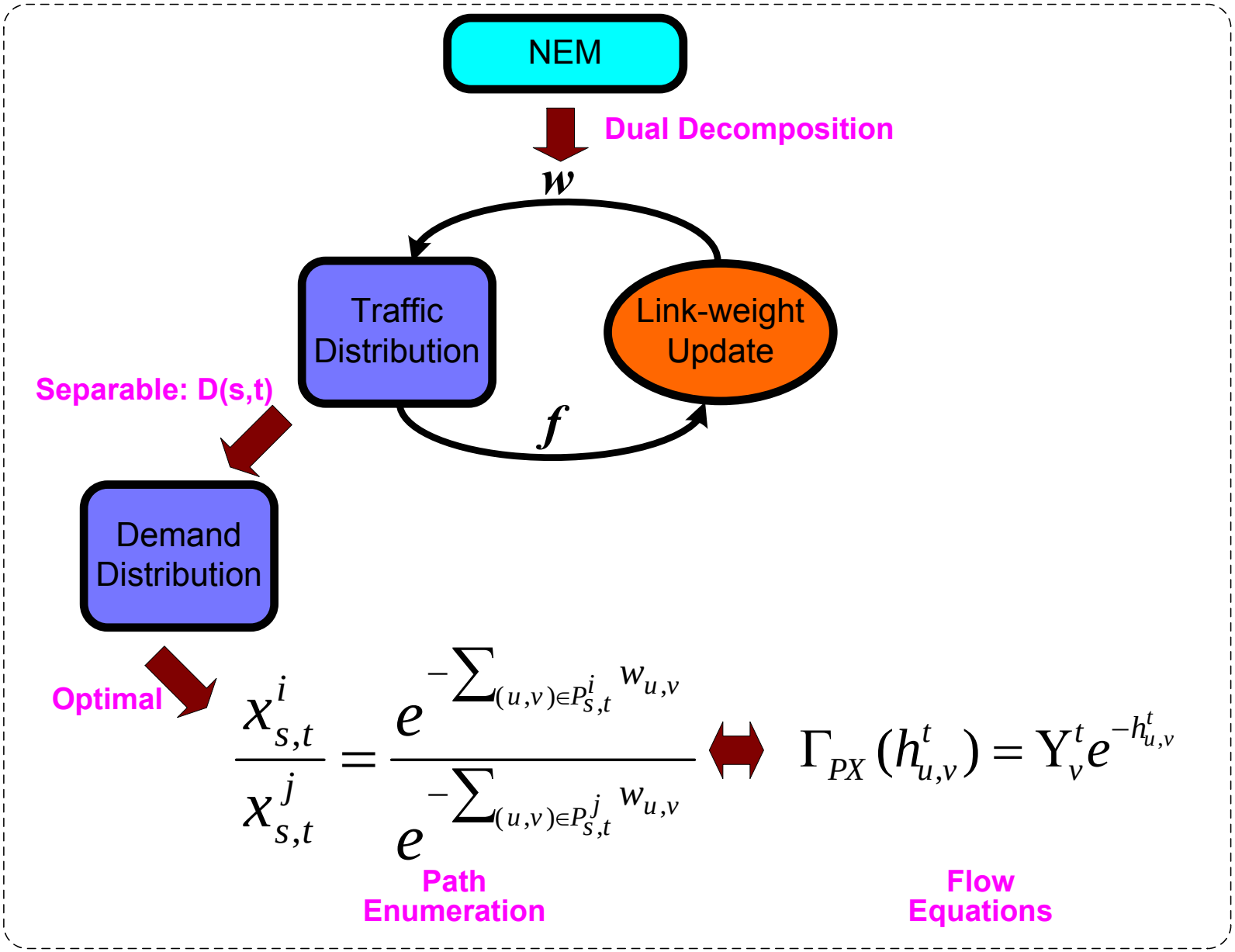
Solve NEM with Dual Decomposition



Solve NEM with Dual Decomposition



Solve NEM with Dual Decomposition



Outline

Outline

Why Entropy?

- Statistical mechanics: many microscopic behaviors aggregate into macroscopic states
- An isolated thermodynamic system eventually reaches an equilibrium macroscopic state that is the most likely one.

Entropy for IP routing

- Consider a network, one source-destination (s, t), P un-capacitated paths and T packets. Randomly route each packet.
- Microscopic behavior: a collection of routing decisions, total P^T
- Macroscopic state: $T_i \geq 0$ packets on path i, with $\sum_i T_i = T$, corresponding to $K = \frac{T!}{\prod_i T_i!}$ microscopic behaviors

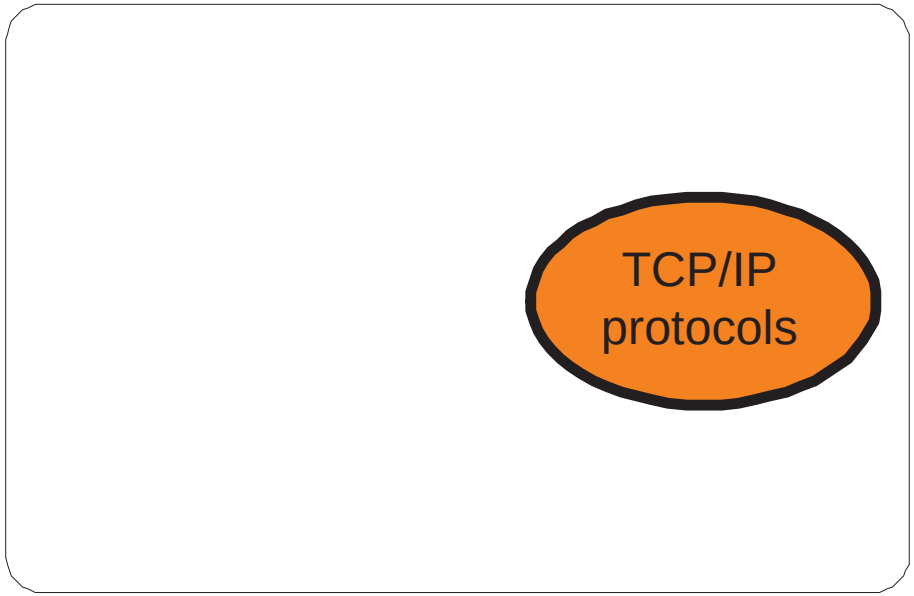
Entropy for IP routing

- Consider a network, one source-destination (s, t), P un-capacitated paths and T packets. Randomly route each packet.
- Microscopic behavior: a collection of routing decisions, total P^T
- Macroscopic state: $T_i \geq 0$ packets on path i, with $\sum_i T_i = T$, corresponding to $K = \frac{T!}{\prod_i T_i!}$ microscopic behaviors
- Stirling's approximation: $\log K \approx -T \sum_i \frac{T_i}{T} \log \frac{T_i}{T}$
- Most likely random routing configuration: $x_i = \frac{T_i}{T}$, fraction of flow on path i.

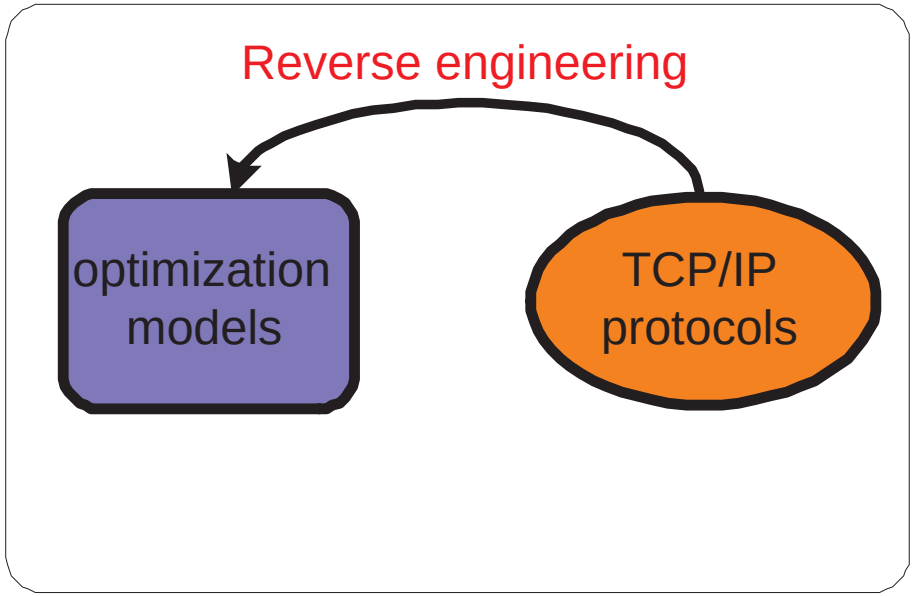
$$\max -\sum_i T x_i \log x_i : \qquad \text{Entropy Maximization}$$

Outline

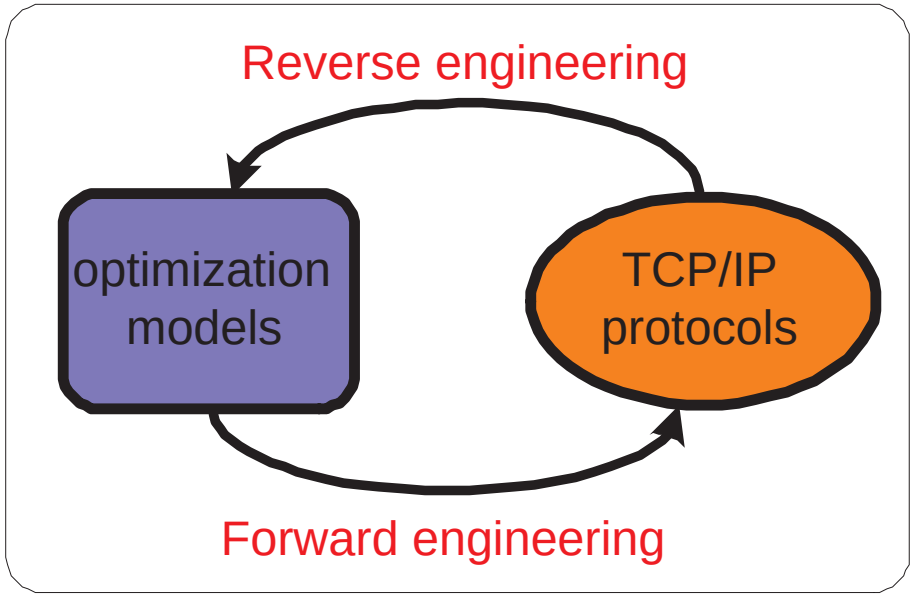
Internet Protocols and Optimization Models



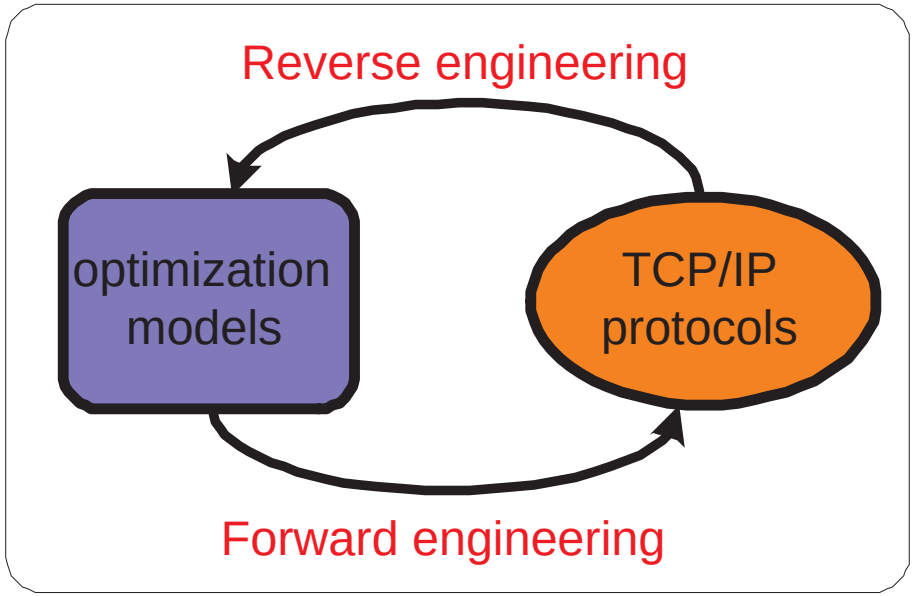
Internet Protocols and Optimization Models



Internet Protocols and Optimization Models

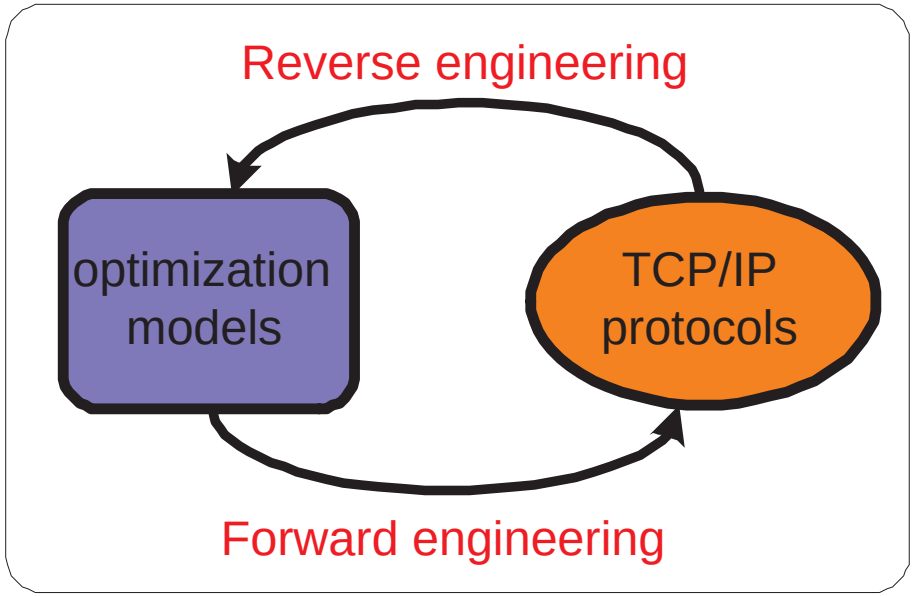


Internet Protocols and Optimization Models



- TCP Congestion Control
  - 1995, S. Shenker, utility on source rate
  - 1998, F. Kelly, network utility maximization
  - 1999-2002, S. Low, R. Srikant, reverse engineering for Reno, Vegas TCP
  - 2002-2005, Forward engineering, FAST TCP, etc.

Internet Protocols and Optimization Models



- TCP Congestion Control
  - 1995, S. Shenker, utility on source rate
  - 1998, F. Kelly, network utility maximization
  - 1999-2002, S. Low, R. Srikant, reverse engineering for Reno, Vegas TCP
  - 2002-2005, Forward engineering, FAST TCP, etc.
- Traffic Engineering for IP?

Traffic Management in TCP vs. IP layer

Property	Congestion Control (TCP)	Traffic Engineering (IP)
Traffic type	Elastic	Inelastic
Flow distribution	Fixed	Variable
Participants	End user and router	Operator and router
Timescale	Seconds	Hours
Optimization Model	Network Utility Maximization	Network Entropy Maximization
Reverse engineering	Tahoe, Reno, Vegas, etc.	Even splitting in OSPF
Forward engineering	FAST TCP, etc.	PEFT

NEM Initiates a New Research Area

- TCP/IP interaction
  - Conventional NUM (fixed routing)
  - Extend NUM (commodity-flow)
  - Realize optimal flow with link-state routing

NEM Initiates a New Research Area

- TCP/IP interaction
  - Conventional NUM (fixed routing)
  - Extend NUM (commodity-flow)
  - Realize optimal flow with link-state routing
- Inter-Autonomous System routing
  - Not always choose closest exit point (hot-potato routing)
- Peer-to-peer networks
  - Allocate rates across multiple demanders for one suppliers
- Fast Failure Recovery
  - Do not always resort to shortest paths in case of failure recovery
- City traffic control, balancing content (media) distribution, etc.

Outline

Design For Manageability

- An intuitively meaningful assumption (e.g. even-splitting across shortest paths in OSPF)
  - Impossible to achieve optimal routing
  - Result in an intractable optimization problem
- Managing a network $\Rightarrow$ designing a manageable network
  - Not look for a better heuristics for an existing protocol
  - Change the routing protocol to enable optimal routing and easier optimization problem

Outline

Open Problems

- Computational Complexity of NEM/PEFT: Polynomial?
- Solve NEM/PEFT + COMMODITY problem altogether?
- Whether DEFT can achieve optimal traffic engineering as well?
- ...

More Information

http://www.research.att.com/∼dahaixu