# 5. Advanced Routing Mechanisms II.

PA159: Net-Centric Computing I.

Eva Hladká

Faculty of Informatics Masaryk University

Autumn 2010

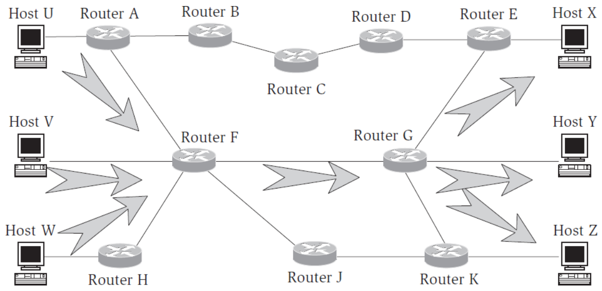
## Lecture Overview I

## Lecture Overview I

# Traffic Engineering in IP Networks

## Introduction I.

- (interior) routing protocols used in IP networks are based on *Shortest Path First (SPF)* routing
- in an unused network, the SPF is ideal:
  - datagrams are delivered expeditiously with the least use of network resources
- **Problem statement:** once traffic increases, a link/router on the shortest path may become saturated
  - while longer paths remain unused/underused
  - *Equal-Cost MultiPath (ECMP)* is usable, but NOT problem-solving solution



## Traffic Engineering in IP Networks

Introduction II.

### Traffic Engineering

**Traffic Engineering** is all about discovering what other paths and links are available in the network, what the current traffic usage is within the network, and directing traffic to routes other than the shortest so that optimal use of the resources in the network is made.

- achieved by a combination of:
  - extensions to existing IGP protocols
  - traffic monitoring tools
  - traffic routing techniques
- occurs *outside* the actual network
- does not address issues such as traffic surge lasting a few seconds/minutes

## Traffic Engineering in IP Networks

### Introduction III.

Performed steps:

1. traffic measurements are collected to estimate the traffic matrix
2. topology and configuration is obtained from the network
3. a link weight determination process determines link weights
    - the computed link weights for each link are injected into the network
        - i.e., each router receives a metrics for its outgoing links
    - once injected, using a normal OSPF/IS-IS flooding process the metrics are disseminated through link-state advertisements

*Question:* How often should the TE system update the link weights?

- up to the network provider/administrator
- usually once a day or once a week
    - to avoid short-term traffic fluctuations
    - since traffic matrix determination is a fairly complex and time-consuming process

## Traffic Engineering in IP Networks

Introduction IV.

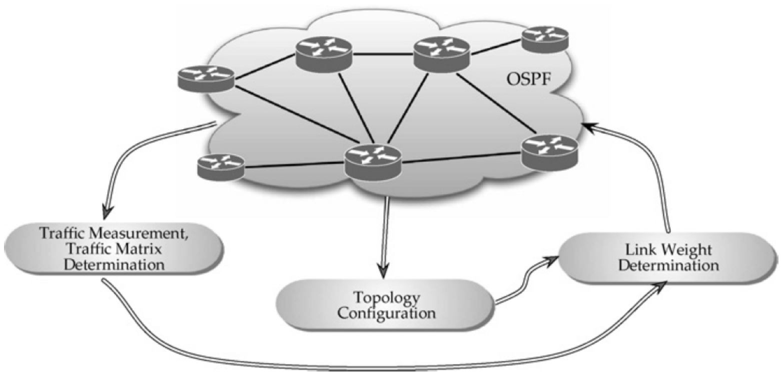
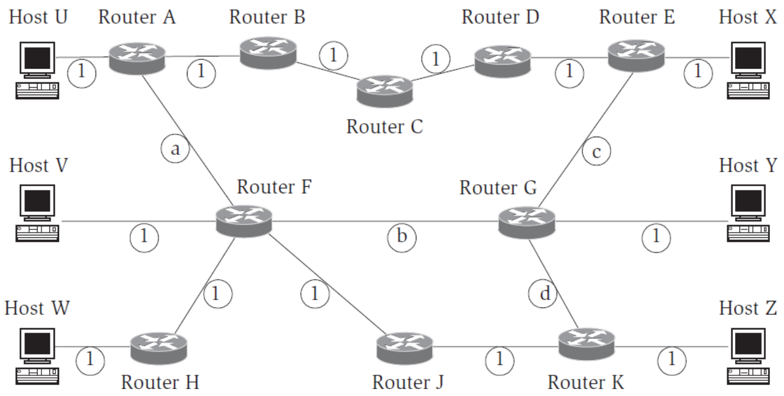


Figure: IP Traffic Engineering architectural framework.

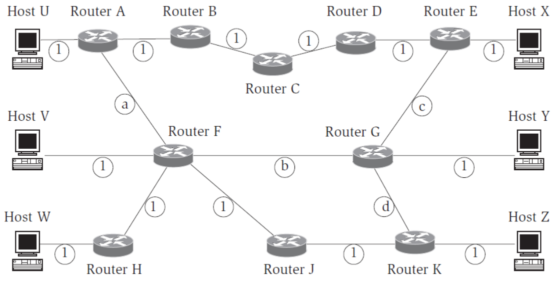
## TE – Complexity illustration I.



*Which costs should be assigned to a, b, c, and d?*

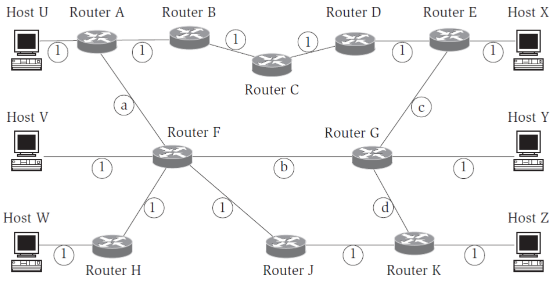
- not trivial even in such a simple network

# TE – Complexity illustration II.



- $a = b = c = d = 1 \Rightarrow$ all traffic tends toward the link FG
- $a = c = d = 1,\ b = 7$
  - $U \rightarrow X$ routed through B, C, D (total cost 6)
  - $W \rightarrow Z$ routed through J (total cost 5)
  - $V \rightarrow Y$ routed through J (total cost 5)
    - not ideal – some congestion is moved to router J

## TE – Complexity illustration III.



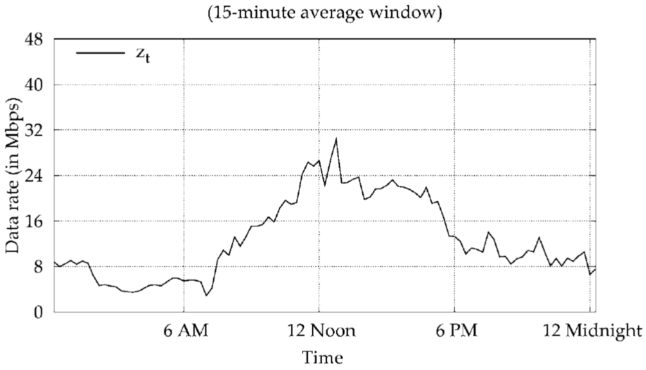
- $a = c = 2$, $b = 7$, $d = 10$ achieves the desired result
  - $U \to X$ routed through B, C, D (cost 6), $V \to Y$ routed through F, G (cost 9), and $W \to Z$ routed through J (cost 5)
  - **But:** imagine $W \to X$ traffic – takes the path WHFABCDEX (cost 9) instead of the shorter path WHFGEX (cost 12)
    - we can increase $a$ to 6
    - but what about $U \to Y$ traffic?
      (will prefer UABCDEGY over shorter UAFGY ☹)

## TE – Discovering Network Utilization I.

- a network-wide view of resource utilization is needed
  - a challenging problem
- several methods to collect and consolidate network usage information exist:
  1. *Simple Network Management Protocol (SNMP)*
     - an application polls each router and converts the returned information into a view of usage across the network
     - does not determine, which flows need to be redistributed to ease any congestion (just an absolute measure of the traffic load is obtained)
  2. *NetFlow*
     - Cisco's tool collecting the information at key points within the network
     - includes aggragation points (*NetFlow collectors*) consolidating the information from a subset of the network
  3. *sFlow*, *ntop*, etc.

## TE – Discovering Network Utilization II.

- network traffic is nonstationary and (usually) time-dependent
  - data rate is different depending on the time of the day
  - ⇒ usually, a *peak* of the traffic data rate (or, say 90% of the peak) over the 24-hour window is considered as a traffic volume needed for traffic engineering considerations

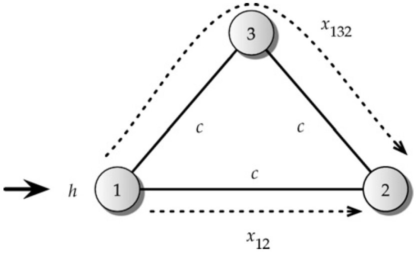


## TE – Discovering Network Topology

- the application making TE decisions must have a clear view of the topology and capabilities of the links within the network
- small, static networks $\Rightarrow$ manual configuration is sufficient
- large and dynamic networks $\Rightarrow$ an automatic system has to be used
  - naturally, extending the IGP routing protocols to distribute additional information about the links will do the job
  - both OSPF and IS-IS have been extended to provide (for each link):
    - traffic engineering metric, maximum bandwidth, maximum reservable bandwidth, unreserved bandwidth, etc.

## Network Flow Modeling – Single-Commodity Network Flow

- *single-commodity* – just a single node pair in the network has positive demand volume
  - commodity ≈ demand for a link's capacity
- let's assume the following network:

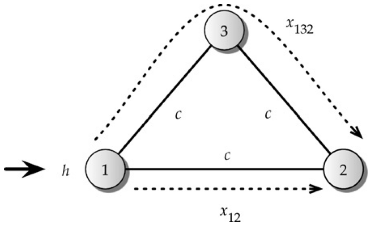


Let's denote:

- $c$ . . . a capacity of each link (here the same for all the links)
- $h$ . . . the demand volume for node pair 1 : 2
- $x_{12}$, $x_{132}$ . . . the amount of the demand volume to be routed over the path $1 - 2$ (resp. $1 - 3 - 2$)

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

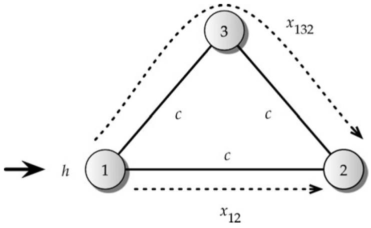


Then the following constraints have to be satisfied:

- the demand volume *h* has to be carried over these two paths:

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

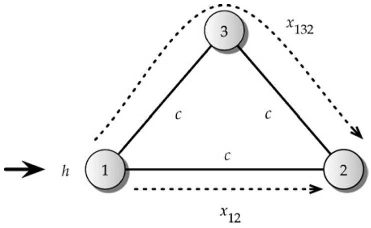


Then the following constraints have to be satisfied:

- the demand volume $h$ has to be carried over these two paths:
  - $\Rightarrow x_{12} + x_{132} = h$

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

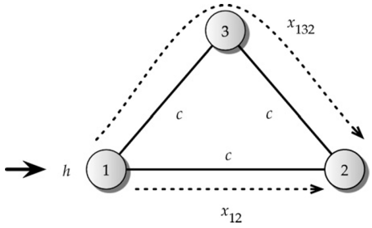


Then the following constraints have to be satisfied:

- the demand volume $h$ has to be carried over these two paths:
  - $\Rightarrow x_{12} + x_{132} = h$
- a path may not carry any negative demand:

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

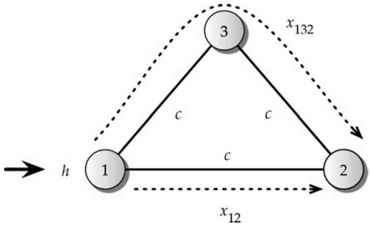


Then the following constraints have to be satisfied:

- the demand volume $h$ has to be carried over these two paths:
  - $\Rightarrow x_{12} + x_{132} = h$
- a path may not carry any negative demand:
  - $\Rightarrow x_{12} \geq 0, \; x_{132} \geq 0$

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

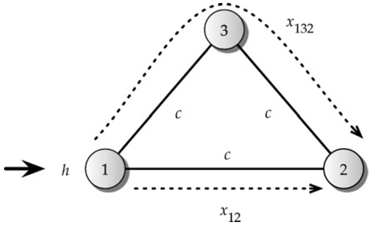


Then the following constraints have to be satisfied:

- the demand volume $h$ has to be carried over these two paths:
  - $\Rightarrow x_{12} + x_{132} = h$
- a path may not carry any negative demand:
  - $\Rightarrow x_{12} \geq 0,\ x_{132} \geq 0$
- any flow on the path cannot exceed the capacity on any of the links the path uses:

## Network Flow Modeling – Single-Commodity Network Flow

### Problem Constraints

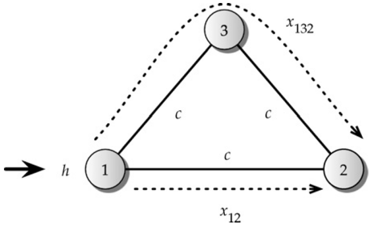


Then the following constraints have to be satisfied:

- the demand volume $h$ has to be carried over these two paths:
    - $\Rightarrow x_{12} + x_{132} = h$
- a path may not carry any negative demand:
    - $\Rightarrow x_{12} \geq 0,\ x_{132} \geq 0$
- any flow on the path cannot exceed the capacity on any of the links the path uses:
    - $\Rightarrow x_{12} \leq c,\ x_{132} \leq c$ (same capacity on each link)

## Network Flow Modeling – Single-Commodity Network Flow

The Goal – Minimize the cost of routing I.

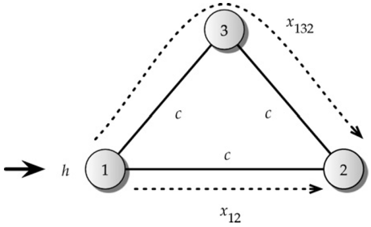


Let's assume **the goal** of minimizing the cost of routing flows:

- let's introduce a cost *per unit* of flow on each path: $\xi_{12}$ and $\xi_{132}$, both $\geq 0$
  - $\approx$ a price payed for data transferred over the path
- $\Rightarrow$ $Total\_cost = \xi_{12}x_{12} + \xi_{132}x_{132}$
  - $=$ the **objective function** (in general denoted by $F$)

## Network Flow Modeling – Single-Commodity Network Flow

### The Goal – Minimize the cost of routing II.



The complete problem could be written as follows:

$$\begin{aligned} \textbf{\textit{minimize}}_{\{x_{12},\,x_{132}\}} \quad & F = \xi_{12}x_{12} + \xi_{132}x_{132} \\ \textbf{\textit{subject to}} \quad & x_{12} + x_{132} = h \\ & x_{12} \le c,\ x_{132} \le c \\ & x_{12} \ge 0,\ x_{132} \ge 0. \end{aligned}$$

*The above system solves a goal of minimizing the cost (price) of routing for the above topology when a traffic demand h is given.*

- it finds proper values of $x_{12}$ and $x_{132}$ satisfying the given conditions

## Network Flow Modeling – Single-Commodity Network Flow

### The Goal – Load Balancing

Another goals could be also considered:

- load balancing – minimization of maximum link utilization
- average delay – minimization of the average packet delay

Example: *Minimization of maximum link utilization:*

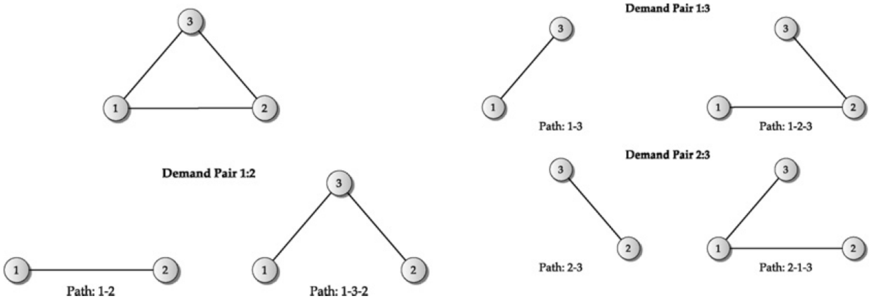
- utilization of the link $1-2$: $\frac{x_{12}}{c}$
- utilization of the links $1-3$ or $3-2$: $\frac{x_{132}}{c}$
- maximum utilization over all links: $max\{\frac{x_{12}}{c}, \frac{x_{132}}{c}\}$

$$\begin{array}{ll} \textbf{\textit{minimize}}_{\{x\}} & F = \max\{\frac{x_{12}}{c},\ \frac{x_{132}}{c}\} \\ \textbf{\textit{subject to}} & x_{12} + x_{132} = h \\ & x_{12} \le c,\ x_{132} \le c \\ & x_{12} \ge 0,\ x_{132} \ge 0. \end{array}$$

*The above system solves a goal of balancing the load over paths* $1-2$ *and* $1-3-2$ *when a traffic demand h is given.*

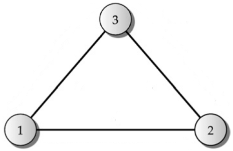
## Network Flow Modeling – Multicommodity Network Flow

- *multicommodity* – all the three demand pairs can have positive demand volumes
  - $h_{12}$, $h_{13}$, $h_{23}$
- for each demand pair, the volume of demand can be accommodated using two paths:



## Network Flow Modeling – Multicommodity Network Flow

### Problem Constraints

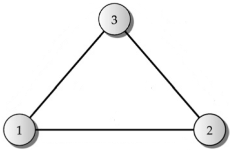


Then the following constraints have to be satisfied:

- the demand volume for each node pair may be carried over two paths:

## Network Flow Modeling – Multicommodity Network Flow

### Problem Constraints

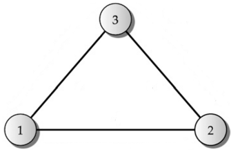


Then the following constraints have to be satisfied:

- the demand volume for each node pair may be carried over two paths:
  - $\Rightarrow x_{12} + x_{132} = h_{12}$
  - $\Rightarrow x_{13} + x_{123} = h_{13}$
  - $\Rightarrow x_{23} + x_{213} = h_{23}$

## Network Flow Modeling – Multicommodity Network Flow

### Problem Constraints

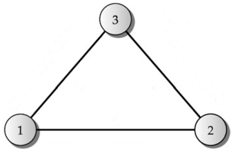


Then the following constraints have to be satisfied:

- the demand volume for each node pair may be carried over two paths:
  - $\Rightarrow x_{12} + x_{132} = h_{12}$
  - $\Rightarrow x_{13} + x_{123} = h_{13}$
  - $\Rightarrow x_{23} + x_{213} = h_{23}$
- links' capacity limits must also be satisfied:

## Network Flow Modeling – Multicommodity Network Flow

### Problem Constraints

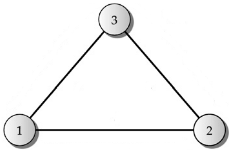


Then the following constraints have to be satisfied:

- the demand volume for each node pair may be carried over two paths:
  - $\Rightarrow x_{12} + x_{132} = h_{12}$
  - $\Rightarrow x_{13} + x_{123} = h_{13}$
  - $\Rightarrow x_{23} + x_{213} = h_{23}$
- links' capacity limits must also be satisfied:
  - $\Rightarrow x_{12} + x_{123} + x_{213} \leq c_{12}$
  - $\Rightarrow x_{13} + x_{132} + x_{213} \leq c_{13}$
  - $\Rightarrow x_{23} + x_{132} + x_{123} \leq c_{23}$

## Network Flow Modeling – Multicommodity Network Flow

### Problem Constraints

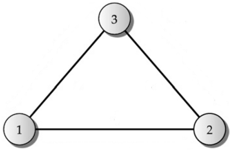


Then the following constraints have to be satisfied:

- the demand volume for each node pair may be carried over two paths:
  - $\Rightarrow x_{12} + x_{132} = h_{12}$
  - $\Rightarrow x_{13} + x_{123} = h_{13}$
  - $\Rightarrow x_{23} + x_{213} = h_{23}$
- links' capacity limits must also be satisfied:
  - $\Rightarrow x_{12} + x_{123} + x_{213} \leq c_{12}$
  - $\Rightarrow x_{13} + x_{132} + x_{213} \leq c_{13}$
  - $\Rightarrow x_{23} + x_{132} + x_{123} \leq c_{23}$
- Total cost:
  - $\Rightarrow Total\_cost = \xi_{12}x_{12} + \xi_{132}x_{132} + \xi_{13}x_{13} + \xi_{123}x_{123} + \xi_{23}x_{23} + \xi_{213}x_{213}$

## Network Flow Modeling – Multicommodity Network Flow

### The Goal – Minimize the cost of routing



Then, **the goal** of minimizing the cost of routing can be formulated as follows:

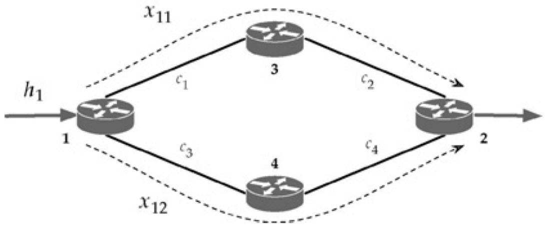
$$\begin{aligned}
&\textbf{\textit{minimize}}_{\{x\}} && F = \xi_{12}x_{12} + \xi_{132}x_{132} + \xi_{13}x_{13} + \xi_{123}x_{123} + \xi_{23}x_{23} + \xi_{213}x_{213} \\
&\textbf{\textit{subject to}} && x_{12} + x_{132} = h_{12} \\
& && x_{13} + x_{123} = h_{13} \\
& && x_{23} + x_{213} = h_{23} \\
& && x_{12} + x_{123} + x_{213} \le c_{12} \\
& && x_{13} + x_{132} + x_{213} \le c_{13} \\
& && x_{23} + x_{132} + x_{123} \le c_{23} \\
& && x_{12} \ge 0,\ x_{132} \ge 0,\ x_{13} \ge 0,\ x_{123} \ge 0,\ x_{23} \ge 0,\ x_{213} \ge 0.
\end{aligned}$$

*(Another goals (load balancing, average delay, etc.) can be formulated as well.)*

## TE – Shortest Path Routing and Network Flow

- in an IP network based on OSPF or IS-IS, the shortest paths are computed based on **links' weights**
  - this computation does NOT consider traffic volume or (usually) capacity of the network
  - the previous examples did NOT consider the links' weights

- *How is the shortest path routing related to network flow modeling?*
  - link weights drive the flows
  - let's denote $w$ to be an array of link weights of all links in the network
    - $w = (w_1, w_2, w_3, \dots)$
    - a dependency of a flow $x_{11}$ on the link weights will be denoted as $x_{11}(w)$



## TE – Shortest Path Routing and Network Flow

### MCSPRF optimization problem I.

**The goal:** *to determine link weights for given traffic volume demand and capacity limits where a certain objective is optimized.*

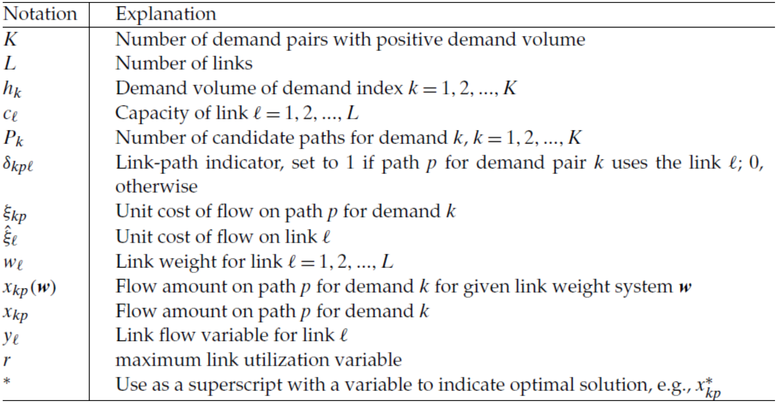
The *Multicommodity shortest path-based routing flow (MCSPRF)* optimization problem having the objective to *minimize the maximum link utilization (load balancing)* can be formulated as follows:

$$\begin{array}{lll}
\textit{minimize}_{\{\boldsymbol{w},r\}} & F = r & \\
\textit{subject to} & \displaystyle\sum_{p=1}^{P_k} x_{kp}(\boldsymbol{w}) = h_k, & k = 1, 2, \ldots, K \\
& \displaystyle\sum_{k=1}^{K}\sum_{p=1}^{P_k} \delta_{kp\ell}\, x_{kp}(\boldsymbol{w}) = y_\ell, & \ell = 1, 2, \ldots, L \\
& y_\ell \le c_\ell\, r, & \ell = 1, 2, \ldots, L \\
& w_1, w_2, \ldots., w_L \in \mathcal{W} & \\
& x_{kp}(\boldsymbol{w}) \ge 0, & p = 1, 2, \ldots, P_k, \quad k = 1, 2, \ldots, K \\
& y_\ell \ge 0, & \ell = 1, 2, \ldots, L \\
& r \ge 0. &
\end{array}$$

## TE – Shortest Path Routing and Network Flow

### MCSPRF optimization problem II.

Where



| Notation | Explanation |
|---|---|
| $K$ | Number of demand pairs with positive demand volume |
| $L$ | Number of links |
| $h_k$ | Demand volume of demand index $k = 1, 2, ..., K$ |
| $c_\ell$ | Capacity of link $\ell = 1, 2, ..., L$ |
| $P_k$ | Number of candidate paths for demand $k$, $k = 1, 2, ..., K$ |
| $\delta_{kp\ell}$ | Link-path indicator, set to 1 if path $p$ for demand pair $k$ uses the link $\ell$; 0, otherwise |
| $\xi_{kp}$ | Unit cost of flow on path $p$ for demand $k$ |
| $\hat{\xi}_\ell$ | Unit cost of flow on link $\ell$ |
| $w_\ell$ | Link weight for link $\ell = 1, 2, ..., L$ |
| $x_{kp}(\boldsymbol{w})$ | Flow amount on path $p$ for demand $k$ for given link weight system $\boldsymbol{w}$ |
| $x_{kp}$ | Flow amount on path $p$ for demand $k$ |
| $y_\ell$ | Link flow variable for link $\ell$ |
| $r$ | maximum link utilization variable |
| $*$ | Use as a superscript with a variable to indicate optimal solution, e.g., $x^*_{kp}$ |

The weights are determined by solving a *dual problem*.

- *details:* PA163: Constraint programming (dr. Rudová)

## TE – Shortest Path Routing and Network Flow

### MCSPRF – Minimum cost objective

$$
\begin{array}{lll}
\boldsymbol{minimize}_{\{\boldsymbol{w}\}} & F = \sum_{k=1}^{K} \sum_{p=1}^{P_k} \xi_{kp} x_{kp}(\boldsymbol{w}) & \\
\boldsymbol{subject\ to} & \sum_{p=1}^{P_k} x_{kp}(\boldsymbol{w}) = h_k, & k = 1, 2, ..., K \\
& \sum_{k=1}^{K} \sum_{p=1}^{P_k} \delta_{kp\ell} x_{kp}(\boldsymbol{w}) \le c_\ell, & \ell = 1, 2, ..., L \\
& w_1, w_2, ..., w_L \in \mathcal{W} & \\
& x_{kp}(\boldsymbol{w}) \ge 0, & p = 1, 2, ..., P_k, \quad k = 1, 2, ..., K.
\end{array}
$$

## TE – Shortest Path Routing and Network Flow

MCSPRF – Minimum cost AND load balancing objective

$$
\begin{array}{lll}
\textbf{\textit{minimize}}_{\{x,r\}} & F = \alpha \sum_{k=1}^{K} \sum_{p=1}^{P_k} \left( \sum_{\ell=1}^{L} \hat{\xi}_\ell \delta_{kp\ell} \right) x_{kp} + \beta r & \\
\textbf{\textit{subject to}} & \sum_{p=1}^{P_k} x_{kp} = h_k, & k = 1, 2, ..., K \\
 & -\sum_{k=1}^{K} \sum_{p=1}^{P_k} \delta_{kp\ell} x_{kp} + c_\ell r \geq 0, & \ell = 1, 2, ..., L \\
 & x_{kp} \geq 0, & p = 1, 2, ..., P_k, \quad k = 1, 2, ..., K. \\
 & r \geq 0. &
\end{array}
$$

## Lecture Overview I

## Multiprotocol Label Switching (MPLS)

### Introduction I.

*Multiprotocol Label Switching (MPLS)*

- a new forwarding mechanism originally presented as a way of improving the forwarding speed of core IP routers
- in MPLS network, packets are forwarded based on *labels*
    - a label is added in front of a packet (i.e., as another header so that routers know how to act based on this label)
        - assigned when packet enters the MPLS-capable network
    - internal MPLS routers don't inspect packet's IP address
        - short and fixed-length label lookup is much faster than longest-prefix match performed on every router
    - labels usually correspond to IP destination networks
        - but can also correspond to other parameters, such as QoS or source address
    - requires new protocols to distribute label information
        - or extensions to existing protocols

## Multiprotocol Label Switching (MPLS)

### Introduction II.

*Multiprotocol Label Switching (MPLS) – cont'd.*

- MPLS flows are *connection-oriented* and packets are routed along pre-configured *Label Switched Paths (LSPs)*
  - the MPLS connection (LSP) is *unidirectional*
  - ⇒ two-way communication requires a pair of LSPs to be established
    - the paths for forward and reverse directions may differ
- MPLS allows new forwarding paradigms not available with conventional IP routing
  - e.g., the ability of network operators to dictate the path that traffic takes through their network, Virtual Private Network support, etc.
    - for example, low-priority data may be sent on a longer path to keep the shortest path clear for higher-priority traffic
- MPLS has emerged into a crucial standard technology for large-scale IP networks

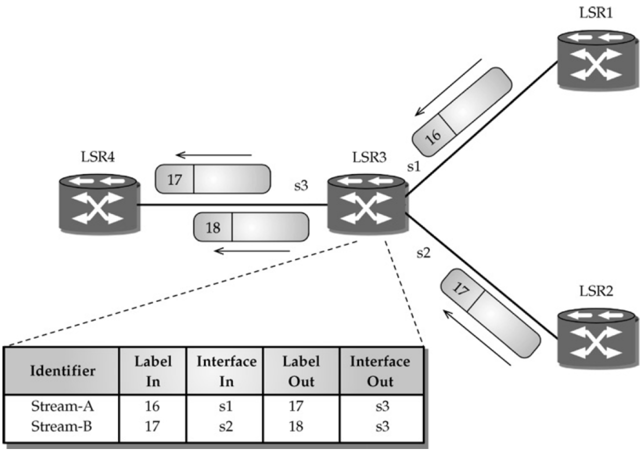
## Multiprotocol Label Switching (MPLS)

### Basic functionality

- an analysis of packets entering the network
    - and their classification to *FEC classes (Forward Equivalence Class)*
        - the classification may be based on more information than just on the destination address
        - for example, type of service, VPN, etc.
- labels' creation for all the FEC classes
- determination/creation of *Label Switched Paths (LSPs)*
- labels' distribution
- setting the forwarding information tables in the routers
    - the tables are known as *Label Information Base (LIB)* or *Label Forwarding Information Base (LFIB)*
    - the tables map $\{incoming\_interface, incoming\_label\}$ to $\{outgoing\_interface, outgoing\_label\}$
        - each MPLS core router maintains a valid mapping from the label of an incoming packet ("incoming label") to a label to be attached to the packet before being sent out ("output label")
- packets' forwarding (based on the label)
- MPLS header (called *shim header*) creation

# Multiprotocol Label Switching (MPLS)

## MPLS Example



| Identifier | Label In | Interface In | Label Out | Interface Out |
|---|---|---|---|---|
| Stream-A | 16 | s1 | 17 | s3 |
| Stream-B | 17 | s2 | 18 | s3 |

Figure: Label swapping and label switched paths.

## Multiprotocol Label Switching (MPLS)

### MPLS Network Components I.

**Edge Label-Switched Routers (Edge-LSRs)** = border routers

- *Ingress-LSR*
  - analyses information in IP packet header
  - based on analysed information, the packet is assigned to particular FEC
  - depending on the assigned FEC, a proper label is inserted into MPLS header
- *Egress-LSR*
  - removes MPLS header and forwards original IP packet to an egress link
  - decrements packet's TTL field

**Core Label-Switched Routers (Core-LSRs)**

- ensures packets' forwarding based on the assigned label
- the IP header is neither modified nor analysed by the Core-LSRs
  - just MPLS labels are analysed and modified, if necessary

## Multiprotocol Label Switching (MPLS)

### MPLS Network Components II.

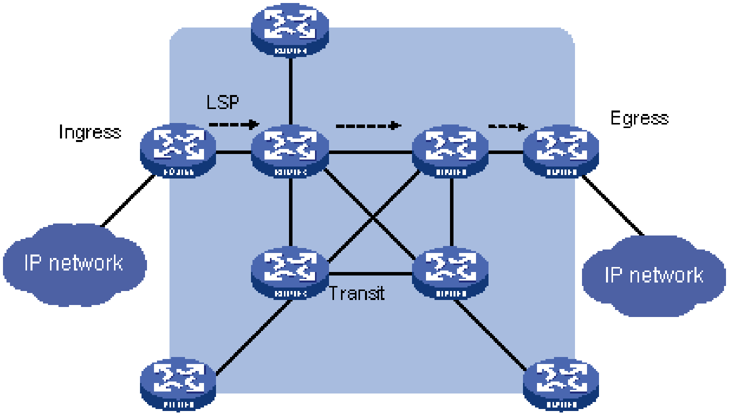
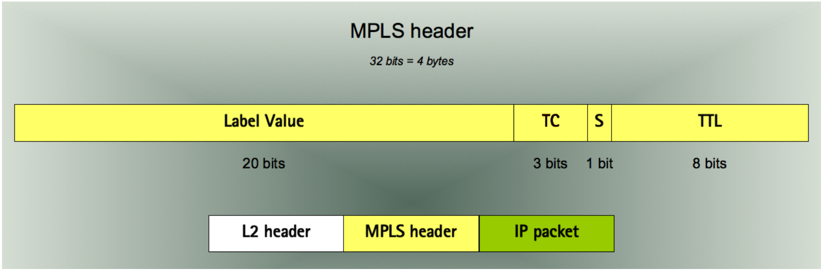


Figure: Structure of the MPLS network.

## Multiprotocol Label Switching (MPLS)

### MPLS Shim Header



- *Label* – carries the actual value of the Label
- *Traffic Class field* – previously named as *Experimental*
- *Stack* – set to one for the last entry in the label stack, and zero for all other label stack entries
    - receiving router examines the top label only
- *TTL* – used to encode a time to live value

## Multiprotocol Label Switching (MPLS)

### MPLS Labels

- usually, just a single MPLS label is assigned to a packet
- scenarios, that may produce more than one label:
    - MPLS VPNs – 2 labels
        - the top label points to the egress router and the second label identifies the VPN
    - MPLS Traffic Engineering – 2 labels
        - the top label points to the endpoint of the traffic engineering tunnel and the second label points to the destination
    - MPLS TE combined with MPLS VPNs – 3 or more labels
    - etc.

# Multiprotocol Label Switching (MPLS)

## MPLS Label Distribution

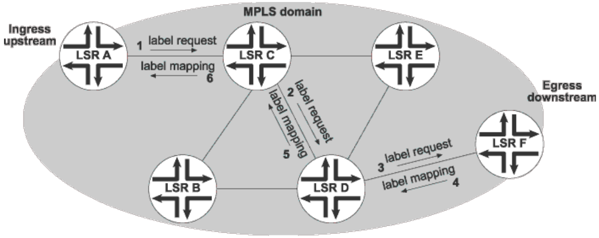
- before an LSP can be used, the LFIBs must be populated at each LSR along the path
  - $\Rightarrow$ a *label distribution protocol* has to be used
- several protocols could be used:
  - *BGP (Border Gateway Protocol)* – its extension allowing labels' distribution
  - *RSVP-TE (RSVP-Traffic Engineering)* – a modified version of the RSVP protocol
  - *LDP (Label Distribution Protocol)* – a specialized protocol for MPLS networks
  - *TDP (Tag Distribution Protocol)* – Cisco's specialized protocol for MPLS networks
  - *LDP/CR (Label Distribution Protocol/Constrained Routing)* – LDP's extension for QoS support
  - etc.

## Multiprotocol Label Switching (MPLS)

### MPLS Label Distribution – Basic approaches I.

*Downstream-on-demand, ordered control approach*

- MPLS devices do not signal a FEC-to-label binding until requested to do so by an upstream device
- an LSR does not advertise a label for a FEC unless it is the egress LSR for the FEC or until it has received a label for the FEC from its downstream peer
- the same label has to be used only between adjacent LSRs!

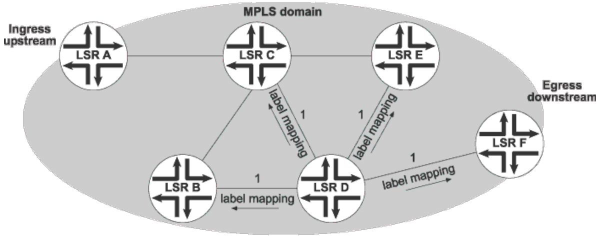


## Multiprotocol Label Switching (MPLS)

MPLS Label Distribution – Basic approaches II.

*Downstream-unsolicited, independent control approach*

- MPLS devices do not wait for a request from an upstream device before signaling FEC-to-label bindings
  - as soon as the LSR learns a route, it sends a binding for that route to all peer LSRs, both upstream and downstream
- the LSR sending the label acts independently of its downstream peer
  - it does not wait for a label from the downstream LSR before it sends a label to its peers



## Multiprotocol Label Switching (MPLS)

### MPLS Label Distribution – LDP protocol

*Label Distribution Protocol (LDP)*

- a protocol defined by the IETF (RFC 5036) for the purpose of distributing labels in an MPLS environment
- relies on the underlying routing information provided by an IGP in order to forward label packets
- makes use of the TCP or UDP transport protocols
- can operate in both Downstream-on-demand and Downstream-unsolicited modes
- main protocol activities:
  - discovery of LDP-capable LSRs that are "adjacent"
    - LDP's *Discovery* message
  - establishment of a control conversation between adjacent LSRs, and negotiation of capabilities and options
    - LDP's *Adjacency* message
  - advertisement of labels
  - withdrawal of labels
    - both performed by LDP's *Label Advertisement* message
  - error notifications
    - LDP's *Notification* message

## Multiprotocol Label Switching (MPLS)

### Traffic Engineering in MPLS I.

MPLS is able to supply much of the function of the traffic engineered overlay model in an integrated manner:

- MPLS has the ability to establish an LSP that follows a path other than the one offered as “preferred” by the routing protocol and forwarding algorithm
- resources within the network can be dynamically reserved as LSPs are established and can be dynamically updated as the needs of the LSPs change
    - traffic flows can be guaranteed a level and quality of service
- traffic can be groomed onto “parallel” LSPs
    - multiple LSPs can be established between a pair of source and destination end points
    - traffic can be distributed over the LSPs by a defined algorithm
- recovery procedures can be defined describing how traffic can be transferred to alternate LSPs in the event of a failure
    - indicating how and when backup and standby LSPs should be set up and routed
- load-sharing and traffic grooming decisions need to be made just once (at the entry point into the LSP) rather than at each node within the network
- etc.

# Multiprotocol Label Switching (MPLS)

## Traffic Engineering in MPLS II.

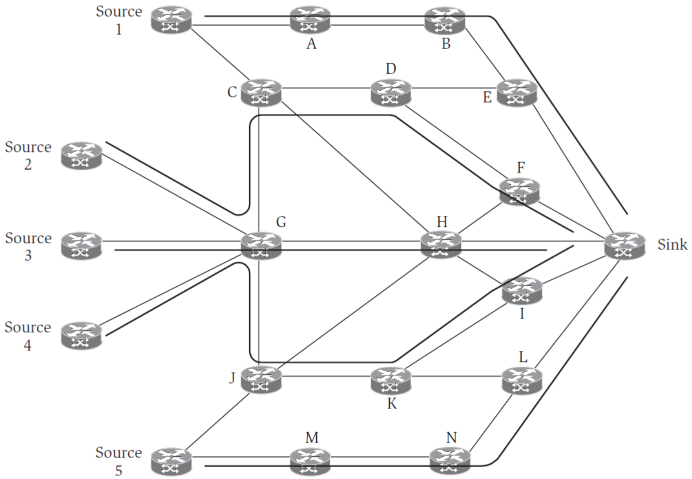


Figure: Explicit path control in an MPLS network.

## Generalized MPLS (GMPLS)

*Generalized MPLS (GMPLS)*

- MPLS has been designed to switch packets using a labeling mechanism
- however, there is the need for an MPLS control-type functionality for controls that is beyond just switching packets
  - e.g., wavelength switching, time division multiplexing, fiber (port) switching, etc.
    - traditionally referred to as *circuit switching* or *circuit routing* (a dedicated path and physical resources must be allocated for a service from one end to another)
- GMPLS thus intended for the following switching capabilities:
  - *Packet-Switch Capable* – (i.e., GMPLS encompasses MPLS)
  - *Time-Division Multiplexing Capable* – for timeslot-based circuit switching
  - *Lambda-Switch Capable* – for wavelength switching at optical cross-connects
  - *Fiber-Switch Capable* – for fiber-level switching at optical cross-connects

## Grid-enabled GMPLS (G$^2$MPLS)

*Grid-enabled GMPLS (G$^2$MPLS)*

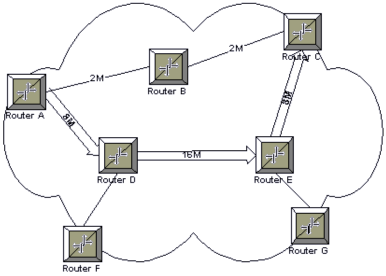
- a network control plane solution that enhances the GMPLS and provides a single-step resource reservation, co-allocation, and maintenance of both network and Grid resources
  - designed by IST *Phosphorus* project
- seamlessly serves Grid jobs by co-allocating and provisioning network and Grid resources in a single-step
- not widely used (yet)

## Lecture Overview I

## QoS-Based Routing – Introduction I.

**QoS-Based Routing** is defined as:

- *a routing mechanism under which paths for flows are determined based on some knowledge of resource availability in the network as well as the QoS requirement of the flows*, or
- *a dynamic routing protocol that has expanded its path-selection criteria to include QoS parameters such as available bandwidth, link and end-to-end path utilization, node resources consumption, delay and latency, and induced jitter*



## QoS-Based Routing – Introduction II.

*Objectives of QoS-based Routing:*

- *to meet the QoS requirements of end users*
  - QoS-based routing is supposed to dynamically find a path from source to destination which can satisfy user's requirements on bandwidth, end-to-end delay, etc.
- *to optimize the network resource usage*
  - QoS-based routing is expected to direct network traffic in an efficient way that can maximize the total network throughput
- *to gracefully degrade network performance when things like congestion happen*
  - when network is in heavy load, QoS-based routing is expected to give better performance (e.g., better throughput) than the best-effort routing, which can degrade the performance dramatically

## QoS-Based Routing – Issues I.

**Metric and path computation**

- How to measure and collect network state information?
- How to compute routes based on the information collected?
- a suitable *metric* has to be chosen (e.g., available bandwidth, delay, jitter, etc.)
- path computation also closely related to resource reservation
  - once a feasible path is chosen, the corresponding resources (bandwidth, buffer space in routers etc.) must be reserved for the traffic flow thus are not available to other flows

**Knowledge propagation and maintenance**

- How often is the routing information exchanged between the routers?
  - more information has to be exchanged than in the case of best-effort routing
    - QoS information (available BW) has to be exchanged along with common routing information like connection topology changes
  - the metrics used by QoS-based routing could be changing very quickly
    - if the routing information is exchanged every time the values of metrics change, it will cause a great burden for the network links and routers $\Rightarrow$ a common way is to set a *threshold* to distinguish significant changes from minor changes (routing information accuracy becomes lower, however)

## QoS-Based Routing – Issues II.

**Scaling by hierarchical aggregation**

- QoS-based routing is expected to be scalable
    - in order to keep the complexity of path computation and the amount of information need to be exchanged and maintained under control, a *hierarchical aggregation* is used
        - however, such aggregation brings inaccuracy in regard of routing information

**Administrative Control**

- different flows in the network should have different priorities
- in the framework having multiple service classes (e.g., DiffServe), the resources should be allocated fairly among all the classes
    - to avoid starvation of lower priority classes

**Integration of QoS-based routing and Best-effort routing**

- for compatibility, QoS-based routing must be able to support best-effort routing
    - i.e., both routing schemes must be able to coexist

## QoS-Based Routing – Routing Algorithms

Basic types

- QoS-based routing algorithms classified according to the way how the state information is maintained and how the search of feasible paths is carried out
  - *source-based routing algorithms*
  - *hop-by-hop routing algorithms* (also called *distributed routing algorithms*)
  - *hierarchical routing algorithms*

## QoS-Based Routing – Routing Algorithms

Basic types – Source-based routing

**Source-based routing algorithms**

- every router has global state information about the network, and the path is locally selected based on the state information
- once the path is determined, the source router notifies the other router along that path how to forward the traffic flow
- *features:*
  - simpler in the sense that it's decided solely by the source
- *drawbacks:*
  - requires that each router has complete state information of the network (hard to maintain)
  - the computation overhead at the source routers is very high
  - $\Rightarrow$ **scalability problems** (not suitable for large networks)

## QoS-Based Routing – Routing Algorithms

Basic types – Hop-by-hop routing

**Hop-by-hop routing algorithms**

- each router just knows the next hop towards the destination
- *features:*
  - used by most current "best-effort" routing protocols ⇒ it's more natural to design and more compatible with existing routing protocols
  - the routing computation burden is distributed among all the routers along the path
- *drawbacks:*
  - it has the routing loop problem (when the routing state information in different routers is not consistent)
  - besides, it also has the scalability problem

## QoS-Based Routing – Routing Algorithms

Basic types – Hierarchical routing I.

**Hierarchical routing algorithms**

- the routing structure consists of multiple levels
  - the bottom level contains the actual routers
  - these routers are organized into some logical groups, which in turn form the *next level*
  - the groups can be further organized into some higher level groups
- the routing information is integrated at the border nodes of each groups
  - every node contains the detailed information about its group and integrated information about other groups
- *features:*
  - scalability $\Rightarrow$ it's suitable for large networks
- *drawbacks:*
  - aggregation decreases the accuracy of the routing state information

## QoS-Based Routing – Routing Algorithms

### Basic types – Hierarchical routing II.

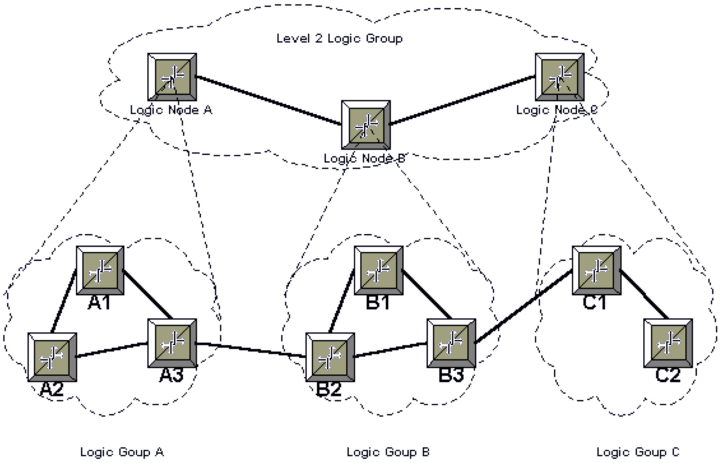


Figure: Hierarchical routing structure.

## QoS-Based Routing – Routing Protocols

PNNI (Private Network-Network Interface)

**Private Network-Network Interface (PNNI)**

- a hierarchical, dynamic routing protocol for ATM networks
- based on link-state algorithm
  - topology information (including information about nodes, links, addresses) is flooded through the network
  - network resources are defined by metrics and attributes (delay, available bandwidth, jitter, etc.)
    - grouped by supported traffic class
  - threshold algorithms are used to determine if the change in a metric or attribute is significant enough to require propagation of updated information
- hierarchical ⇒
  - PNNI has the concepts of levels and logical nodes
  - supports aggregation of topology and reachability information
- *drawbacks:*
  - doesn't support multicast and policy routing, and control of alternate routing
  - inherits the common problem with link state QoS-based routing
    - an issue with efficient broadcast of state information (especially for dynamic metrics)

## QoS-Based Routing – Routing Protocols

QOSPF (QoS routing extensions to OSPF)

**QoS routing extensions to OSPF (QOSPF)**

- QoS extension to OSPF
- also based on link-state algorithm, and is hierarchical protocol
- supposed to be working in an environment in which both QoS-based routing and best-effort routing are needed
- for simplicity, link bandwidth and propagation delay are the only metrics extension added to *Link State Advertisements (LSAs)*
- in order to decrease protocol overhead, LSAs are triggered only when there is a significant change in the value of the metrics since the last advertisement
- a concept of QoS paths pre-computation is used:
    - for every possible destination, the algorithm pre-computes a "widest-shortest path" (a minimum hop count path with maximum bandwidth)