Inside MPLS-TE Routing

Imène CHAIEB, Jean-Louis LE ROUX,

FTR&D/CORE/CPN

15/02/2006

Content

- Context & Objectives
- Preemption Approach
  - ✓ Determining the best LSP order
  - ✓ Appling the order to MPLS-TE networks
- LSP demands splitting
  - ✓ Equal splitting
  - ✓ Unequal splitting

Our main objectives :

- ***Improve optimality and stability of MPLS-TE online uncoordinated mechanisms***

  - Study and suggest solutions which perform well when the network is congested

  - Adapt proactively and or reactively the network to traffic variation or network failure

Preemption Approach

- How to use preemption mechanism to optimize MPLS-TE networks even without failure cases ?
- Preemption priorities to re-order LSPs in uncoordinated mode according to their size
- Two phases :

  (1) Determine which order optimize well the network : Ascending or descending

  (2) Introduce preemption priorities to apply the LSPs sort in the network

Preemption Approach - Phase 1 : Determining the best LSPs order

- Three performances criteria:

  - No blocking cases
    1- Objective function : Network load function

$$\sum_{e \in Network} 1 / CR_e$$

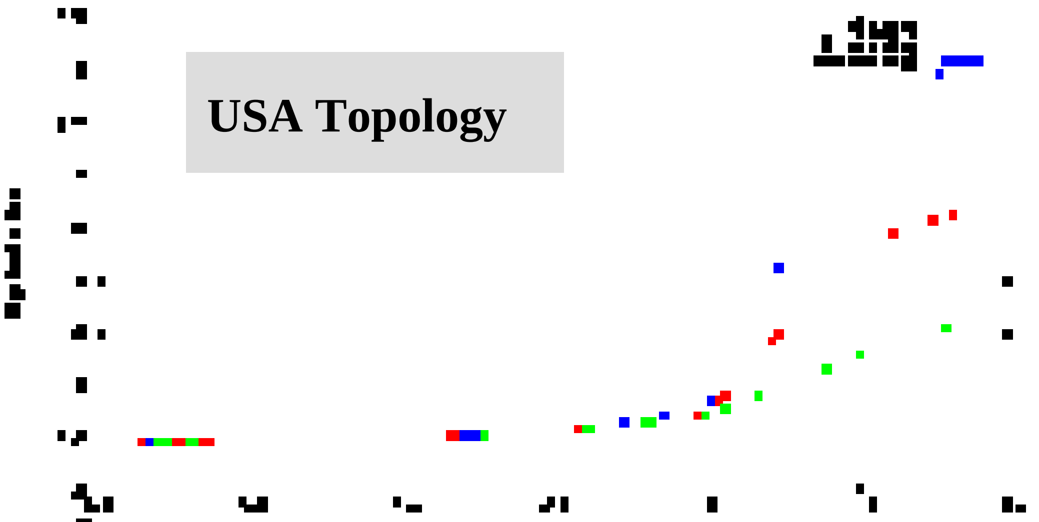
  - Presence of Blocking cases
    – Rejected LSPs ratio
    – Quantity of rejected bandwidth

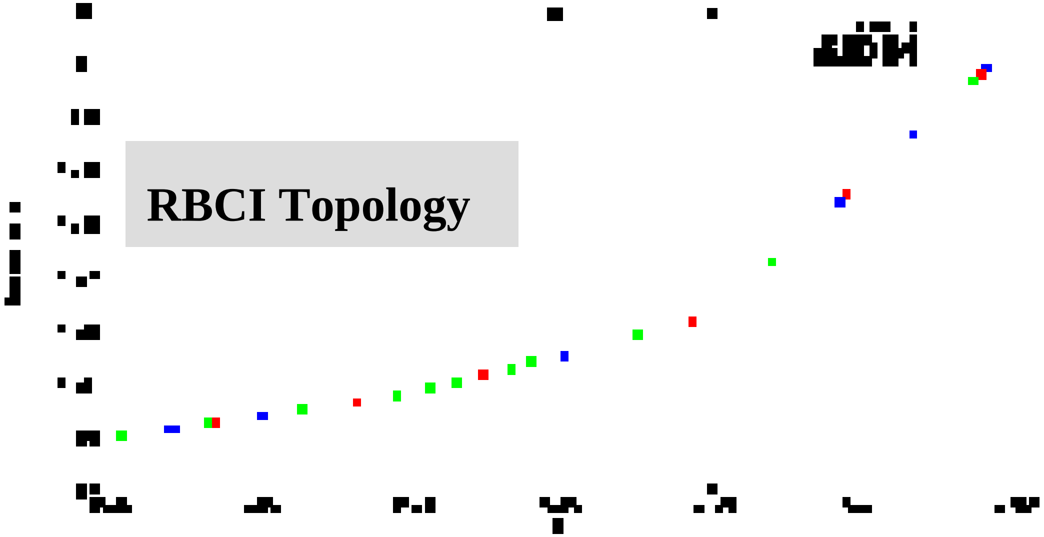
Preemption Approach - Phase 1 : Determining the best LSPs order

- Simulations Assumptions

  - Two topologies :
    - Topology FT network in US :
      - *Network size : 34 nodes (18 edge routers), 112 arcs*
      - *Link capacity : 2,5 G*
      - *Link cost : 10*
      - *We generate, between each edge routers pair,*
    - Topology RBCI :
      - *Network size : 65 nodes , 147 arcs*
    - We vary the bandwidth using a multiplying factor k to increase the network load

Simulations Results (Evaluation criterion = Load Function)

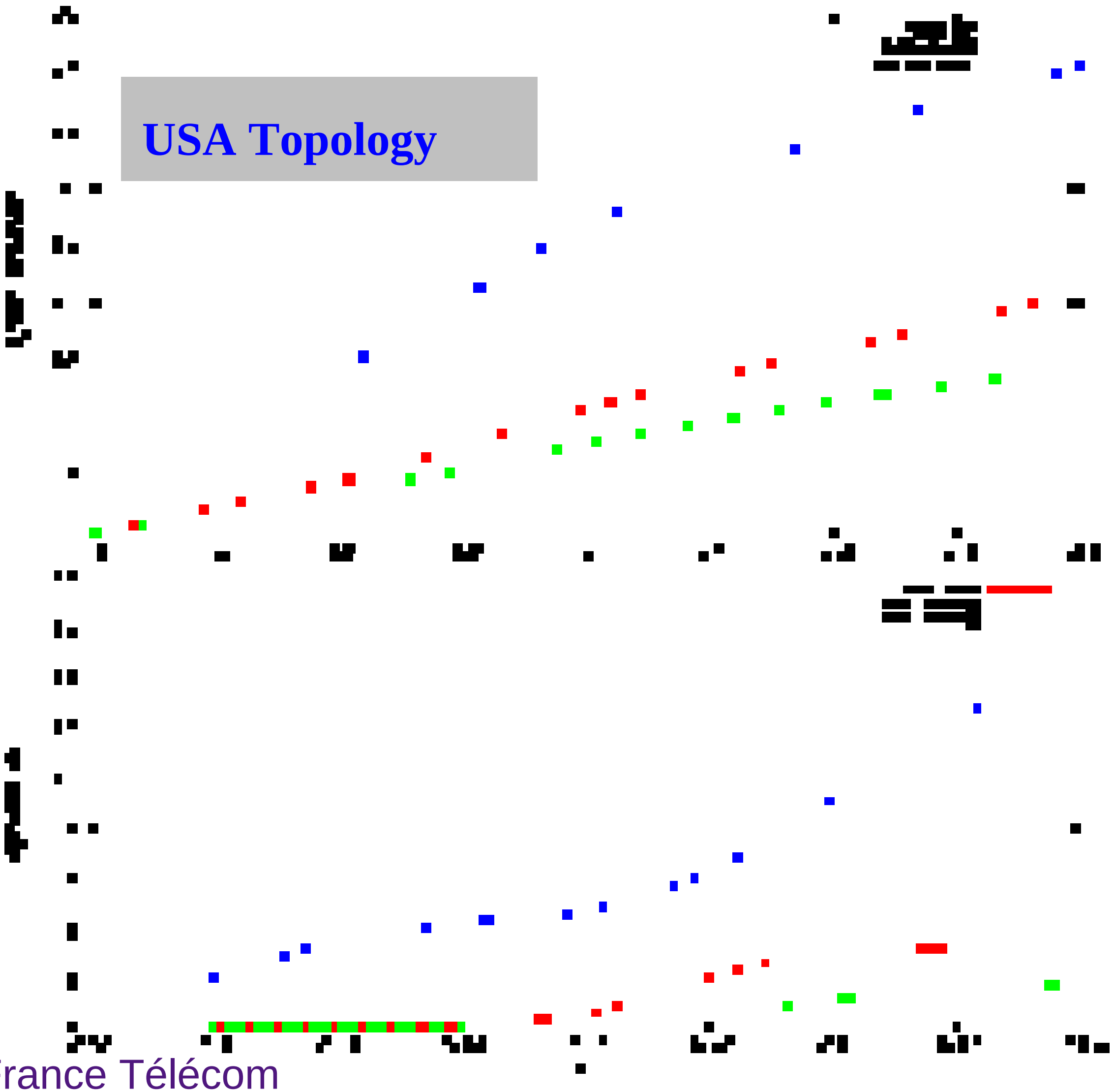




The *increasing* order minimise well the load function

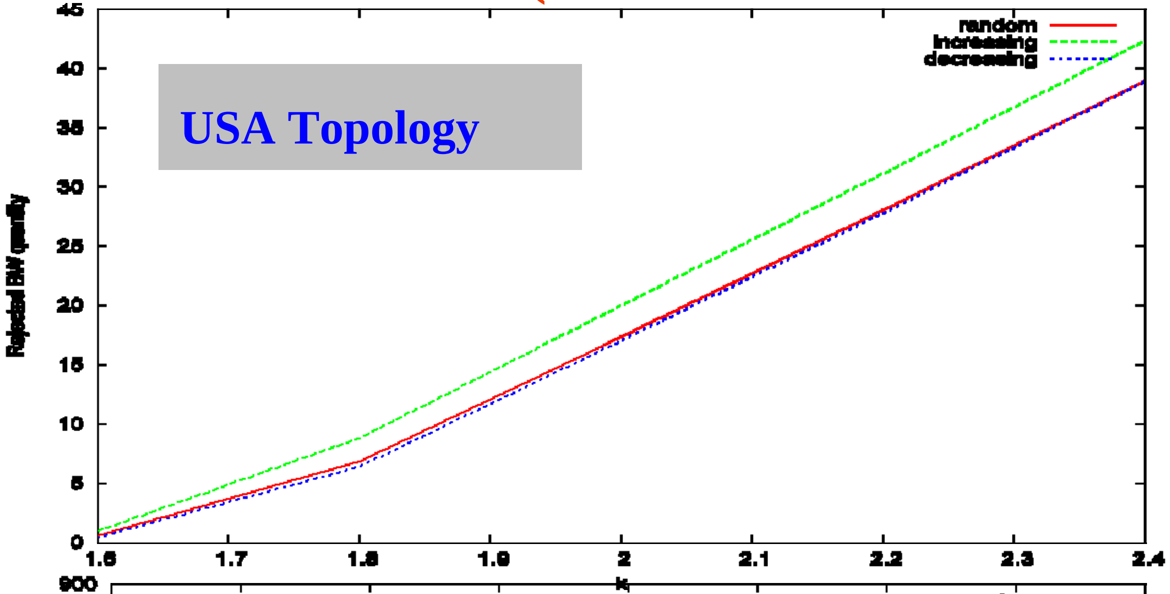
As the RBCI topology is not well meshed, the result is the same !

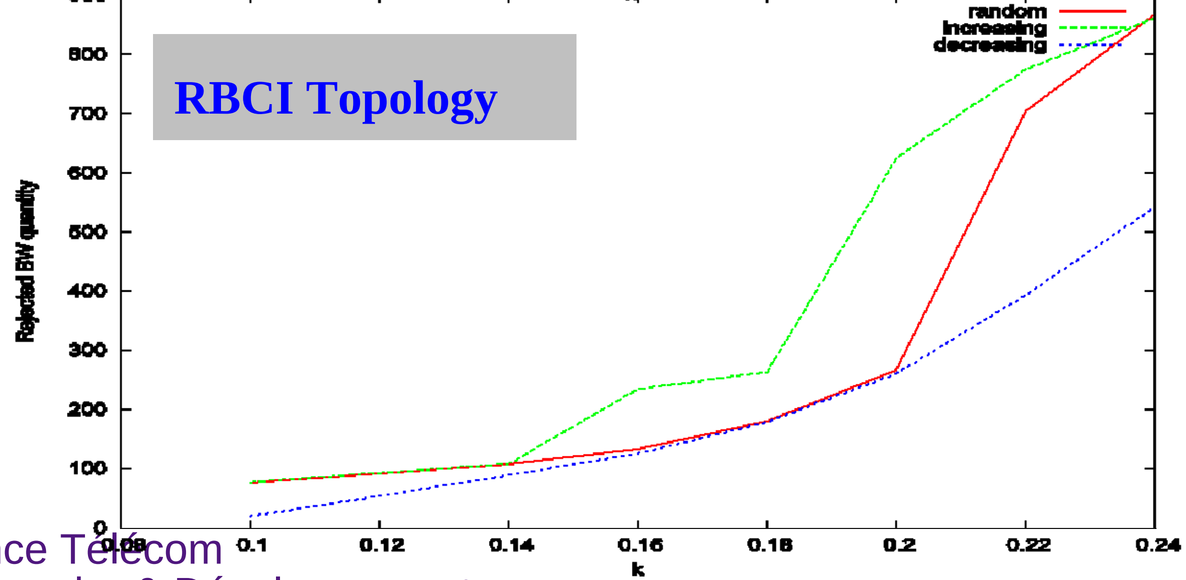
Simulations Results (Evaluation criterion = Rejected LSP ratio)



The *increasing* order minimise well the number of rejected LSPs

Simulations Results (Evaluation criterion = Rejected BW ratio)





The *decreasing* order minimise well the quantity of the rejected bandwidth

Preemption Approach - Phase 2 : Applying preemption priorities

- Problem => 8 priorities for N TE-LSPs (N>8), How to distribute the 8 priorities among the N LSPs ?
- Two methodes :
  - ✓ Linear partitionning
  - ✓ Nonlinear partionning

linear Partitioning (1)

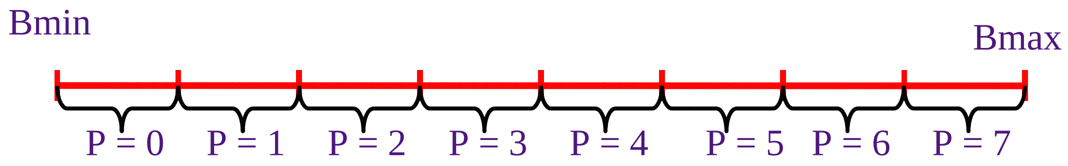
- Divide Bandwidth scale in 8 intervals, the width of each interval is :

$$Bi = (Bmax - Bmin)/8$$

Bmax : The bandwidth of the largest LSP demand
Bmin : The bandwidth of the smallest LSP demand

- Assign a priority *p* to each interval

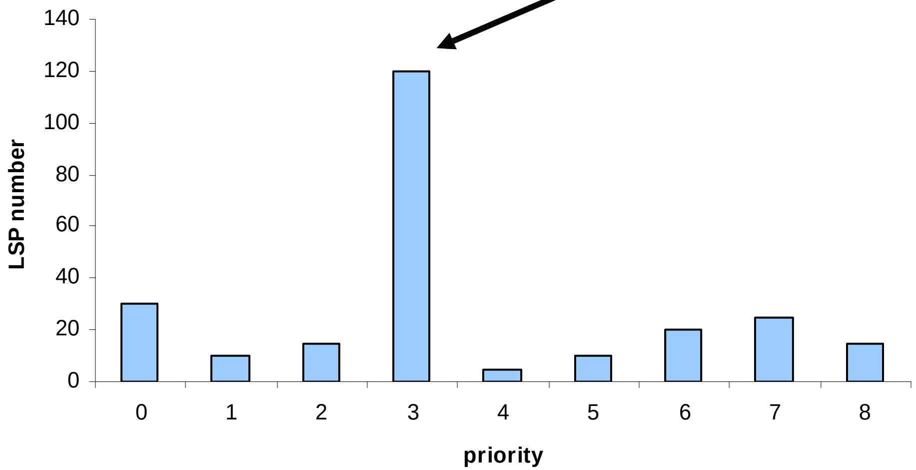
- Assign the same priority to all the LSPs whose their size belongs to the same interval



linear Partitioning (2)

- If the LSPs majority has the same size, they will have the same priority ☹
- The number of LSPs in each interval is not equivalent, the difference can be very significant ☹

120 TE LSPs have the same priority 3: => 57,14 % of all the TE LSP in the network have the same priority



The order is not well applied with the linear approach ☹

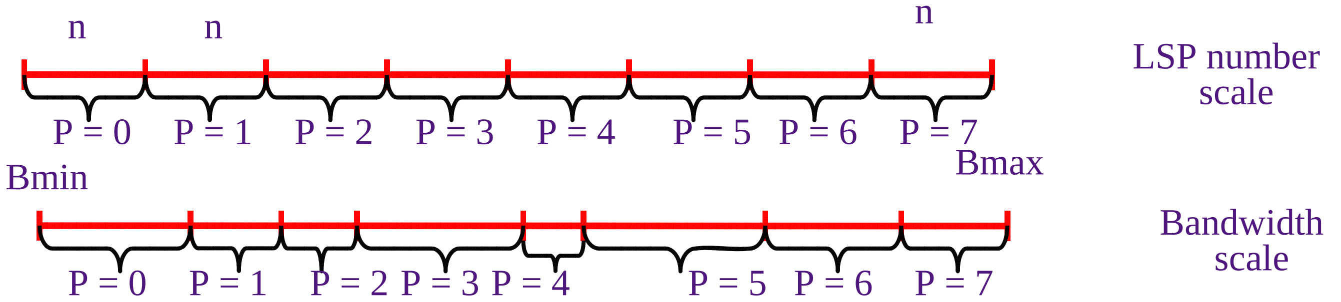
Nonlinear Partitioning (1)

- Take into account the LSP number
- Solution :

  1- Sort the LSP requests (e.g increasing order)

  2- Divide the total number of LSP requests in 8 : n = N/8

  3- Assign to each set of n LSPs a priority

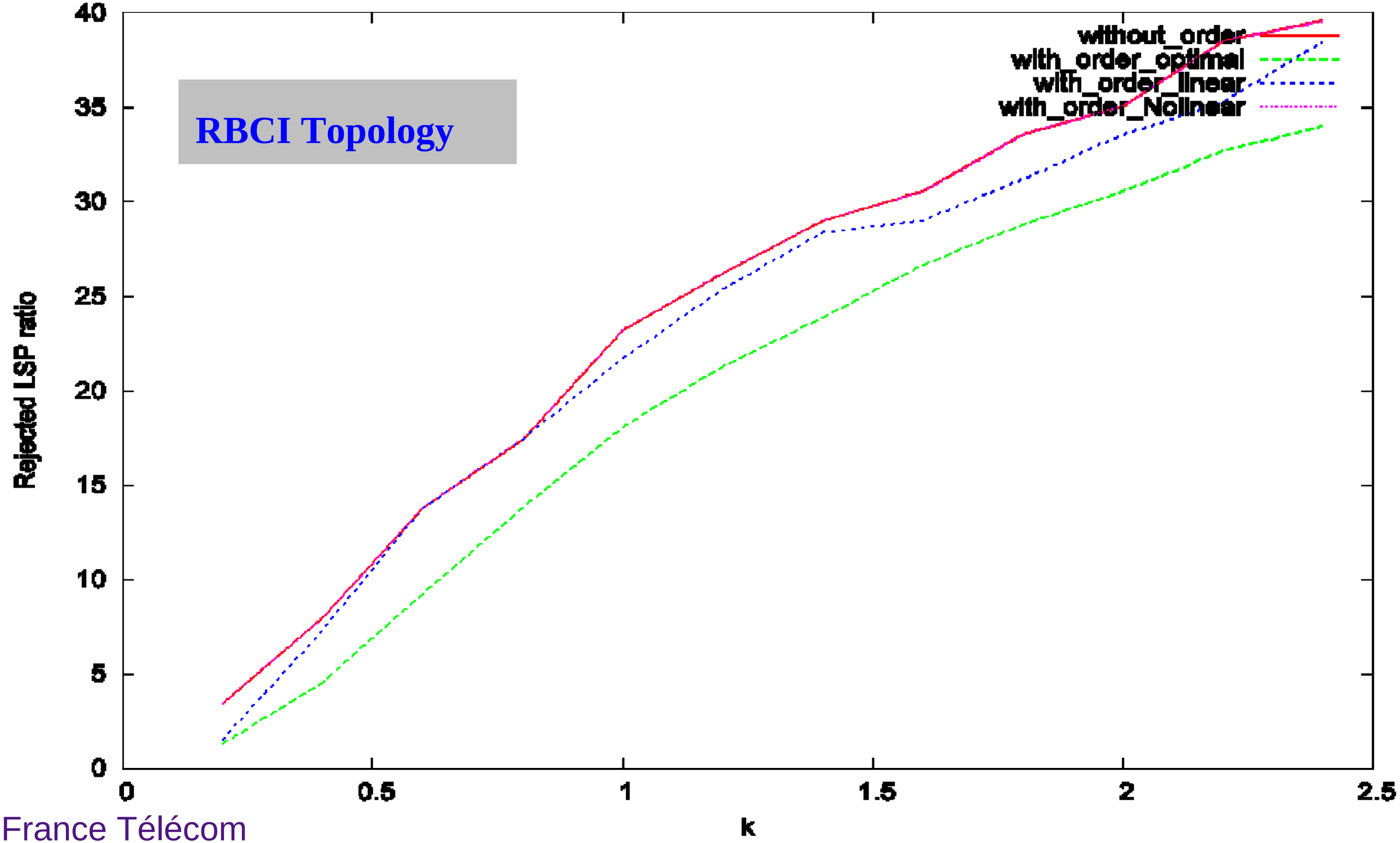


Nonlinear Partitioning (2)

- A major practical problem :

  Ingress Routers must know the total number of requests and the size of each LSPs maintained by all the other ingress routers : It's not possible due to the dynamicity of the network state ☹

Simulations Results : Increasing order



LSP splitting Approach

- How to split a request demand whenever there is no resource to admit it ?
- Several methods can be applied

  1- Equal spitting

  2- Unequal splitting

Equal Traffic splitting (1)

- Step 1 : Apply cspf algorithm to find path for the LSP setup request
- Step 2 : If there is no sufficient resources for the current request j , the demand bandwidth *Bj* is divided in two equal bandwidth demands : Bj/2
- Step 1 is applied to find paths to the two requests
- A threshold for the number of the same demand splitting should be fixed => Find a tradeoff between the optimality and the convergence time.
- Limitation of this approach : It may not converge ☹

Unequal Traffic splitting (1) : Using widest path

A proposal algorithm :

- Step 1 : Apply cspf algorithm to find path for the LSP setup request
- Step 2: Apply the widest path algorithm (WPA) if there is no sufficient resources for the current request j
- Route the fraction of the request that satisfy the widest path
- Apply the step 1 for the reminder fraction of the request, if there is still no sufficient resources, the step 2 is applied
- A threshold of the number of WP computing for the same demand is fixed => Find a tradeoff between the optimality and the convergence time.

Unequal Traffic splitting (2) : Using widest path

- Limitation of this approach : It may block resources for future requests ☹

Unequal Traffic splitting (2) : Using KSP

A proposal algorithm :

- Step 1 : Apply cspf algorithm to find path for the LSP setup request
- Step 2 : If there is no sufficient resources for the current request j (with bandwith Bj), all the links whose capactiy is less than Bj/k are pruned
- Step 3 : Apply the k shortest path algorithm (KSP)
- Step 4: Assyn a *Bj/k* demand to each path of the k-paths.

Conclusion

Next steps…

- Studying more solutions for LSP requests splitting
  - Proactive approaches
- Combining splitting technique and preemption approach
- Comparing with offline coordinated solutions