Advanced Network Design

1 Node location problems

(Pióro - Medhi : 6, 1)

- node location : relevant in long-term planning phase of network design
- node : routers, hubs, switches, ...

Problem :

N : # areas (sources) to be connected

M : # locations candidate to locate "connection" nodes

η_j : cost of location j, if opened, $\forall j = 1, .. M$

ξ_{ij} : cost of connecting area i to location j, if opened, $i = 1 .. N$, $j = 1, .. M$

K_j : maximum number of areas that can be handled at j, $j = 1, .. M$

Constraints: each area needs to be connected to exactly one location node

Objective: minimize the overall cost

Variables

$$x_j = \begin{cases} 1 & \text{if a "connection" node is located at } j \\ 0 & \text{otherwise} \end{cases}$$

location decisions

$$j = 1, \dots M$$

allocation ≡ connection decisions

$$u_{ij} = \begin{cases} 1 & \text{if area } i \text{ is connected to location } j \\ 0 & \text{otherwise} \end{cases}$$

$$i = 1, \dots N, \quad j = 1, \dots M$$

ILP

$$\min \sum_{i=1}^{N} \sum_{j=1}^{M} \xi_{ij} u_{ij} + \sum_{j=1}^{M} \eta_j x_j$$

$$\sum_{j=1}^{M} u_{ij} = 1 \qquad i = 1, \dots N$$

$$\sum_{i=1}^{N} u_{ij} \le K_j x_j \qquad j = 1, \dots M$$

$$u_{ij} \in \{0, 1\} \qquad i = 1, \dots N, \; j = 1, \dots M$$

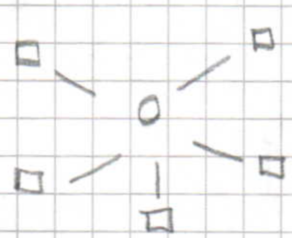
$$x_j \in \{0, 1\} \qquad j = 1, \dots M$$

- This is a NP-Hard 0-1 Linear Programming problem, which can be addressed via the optimization techniques presented in the intermediate part of the course

- Additionally : ad-hoc heuristics

Add Heuristic

- start with any location and all areas connected to this



this may be an infeasible solution (capacity constraints)

- then consider the other locations one at a time, choose the one with the best "savings" and open it to strive feasibility and low cost.

Algorithmic framework

Initialization

- select location $\tilde{j}$; $S_0 = \{\tilde{j}\}$
- F_0 cost of the solution where i is connected to $\tilde{j}$, $\forall i$
- $\tilde{\xi}_i = \xi_{i\tilde{j}}$ $i = 1, .. N$
- $K = 0$

Step 1 $\forall j \in M \setminus S_K$ do * M set of locations *

$$F_j^{K+1} = F^K + \sum_{i \in I_j} (\xi_{ij} - \tilde{\xi}_i) + \eta_j \text{, where}$$

$$I_j = \{ i : \xi_{ij} < \tilde{\xi}_i \}$$

Step 2 • determine a new $\tilde{j}$ such that

$$F_{\tilde{j}}^{K+1} = \min_{j \in M \setminus S_K} \{ F_j^{K+1} \} < F^K !$$

- if there is no such $\tilde{j}$, then go to Step 4

Step 3

- $S_{K+1} = S_K \cup \{\tilde{\delta}\}$
- $\tilde{\xi}_i = \xi_{i\tilde{\delta}} \quad \forall i \in I_{\tilde{\delta}}$
- $F^{K+1} = F^{K+1}_{\tilde{\delta}}$
- $K := K + 1$

go to Step 1

Step 4 no more improvement : STOP

- Greedy heuristic
- Time complexity :

\# iterations $O(M)$

cost per iteration $O(NM)$

$O(NM^2)$ time

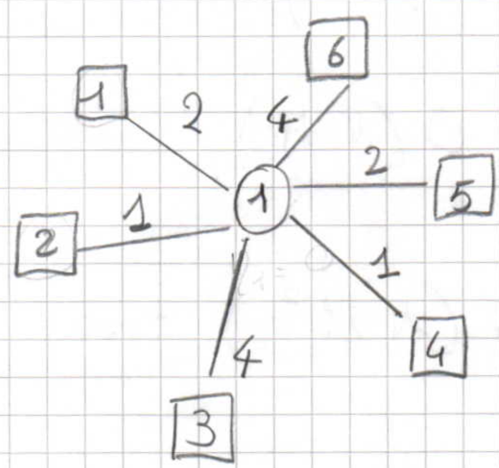
Example (Add heuristic)

- $M = 4 \qquad K_j = 3 \qquad j = 1, .. 4$
- $\eta_1 = 0 \qquad \eta_2 = \eta_3 = \eta_4 = 2$
- $N = 6$

$$\xi_{ij} \quad \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{cccc} 1 & 2 & 3 & 4 \\ \hline 2 & 1 & 2 & 4 \\ 1 & 0 & 1 & 2 \\ 4 & 1 & 2 & 2 \\ 1 & 2 & 1 & 2 \\ 2 & 3 & 2 & 0 \\ 4 & 4 & 3 & 2 \end{array}$$

Initialization

$S_0 = \{1\}$ ("must" location since $\eta_1 = 0$)



$F_0 = 14 + \eta_1 = 14$

< infeasible >

$\tilde{\xi}_i = \xi_{i1} \qquad i = 1, .. 6$

Iteration 1

Step 1 : check one "closed" location at a time to find the one which provides the best savings

- **location 2**

$$(\xi_{12} - \underbrace{\xi_{11}}_{=\xi_1}) \quad (\xi_{22} - \xi_{21}) \quad (\xi_{32} - \xi_{31}) \quad (\xi_{42} - \xi_{41})$$

$$-1 \qquad\qquad -1 \qquad\qquad -3 \qquad\qquad 1$$

$$(\xi_{52} - \xi_{51}) \quad (\xi_{62} - \xi_{61})$$

$$1 \qquad\qquad 0$$

Therefore : $I_2 = \{1, 2, 3\}$

- there would be a good motivation to connect the first three areas to location 2 since this would reduce the connection cost

$$F_2^1 = F^0 + \sum_{i \in I_2} (\xi_{i2} - \xi_{i1}) + \eta_2 =$$

$$= 14 - 5 + 2 = 11$$

- **location 3**

$$I_3 = \{3, 6\}$$

$$F_3^1 = F^0 + \sum_{i \in I_3} (\varrho_{i3} - \varrho_{i1}) + \eta_3 =$$
$$= 14 - 3 + 2 = 13$$

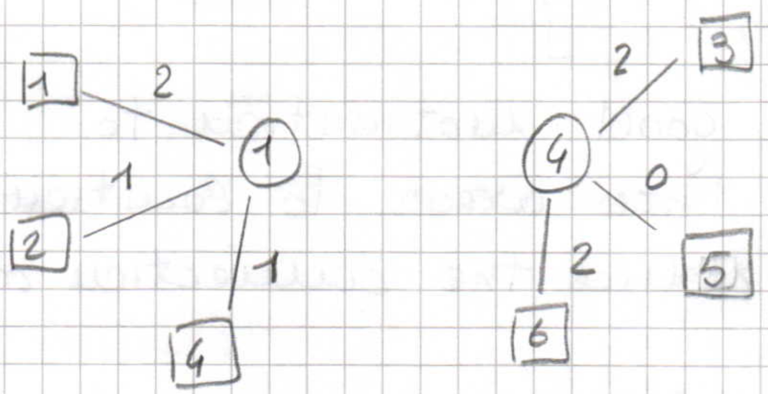
- $\boxed{\text{location } 4}$

$$I_4 = \{3, 5, 6\}$$

$$F_4^1 = F^0 + \sum_{i \in I_4} (\varrho_{i4} - \varrho_{i1}) + \eta_4 =$$
$$= 14 - 6 + 2 = 10$$

Step 2 and Step 3

The cheapest move is to open location 4, by connecting it the areas 3, 5 and 6:



$F^1 = 10$

< feasible >

Iteration 2

Step 1 : check location 2 and location 3 one at a time

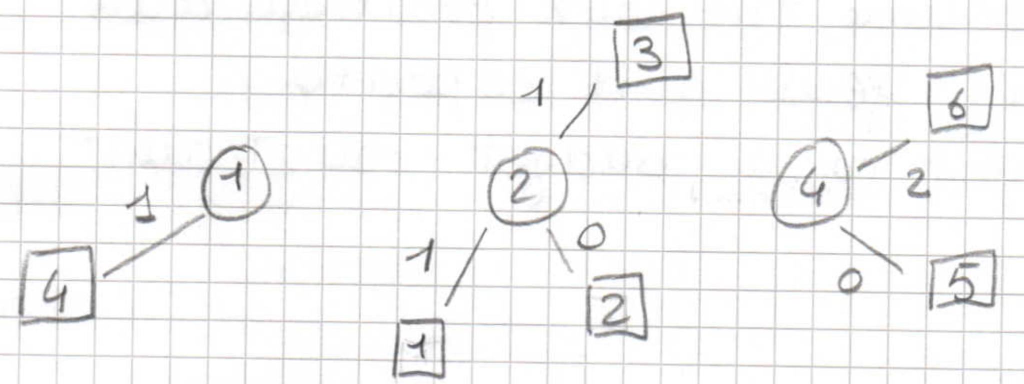
- $\boxed{\text{location } 2}$ $\quad I_2 = \{1, 2, 3\}$

$$F_2^2 = F^1 + \sum_{i \in I_2} (\xi_{i2} - \tilde{\xi}_i) + \eta_2 =$$
$$= 10 - 3 + 2 = 9$$

- $\boxed{\text{location } 3}$ $I_3 = \emptyset$, so no improvement

Step 2 and Step 3

open location 2, and connect areas 1, 2 and 3 to it



$F^2 = 9$

Iteration 3

the only location left to check is 3:
since there is no additional gain: STOP

< return the solution found
at the end of Iteration 2 >

Obs: as presented, the Add Heuristic could return infeasible solutions!

2 Design with budget constraint

(Pióro - Medhi : 6.3.2)

- Topological design problems where traffic demand is taken into account

Assumption : core nodes (access nodes and transit nodes) are already located

Problem : determine the link location and the corresponding flow and capacity allocation subject to a budget constraint

Notation

d : demand between an origin and a destination ≡ commodity

D : set of demands (commodities)

P_d : candidate paths to satisfy d

h_d : volume of demand of d, $\forall d \in D$

c_{ij} : unit cost of link (i,j), $\forall (i,j) \in A$

maintenance cost

[$c_{ij} \rightarrow \xi_e$ in P-M, with $e = (i,j)$]

K_{ij} : cost of installing (i,j), $\forall (i,j) \in A$
(for capital budget)

u_{ij} : upper bound for the capacity of (i,j), $\forall (i,j) \in A$

B : capital budget

Variables

- x_{dp} : flow on path $p \in P_d$, $\forall d \in D$, $p \in P_d$
- y_{ij} : capacity on link (i,j), $\forall (i,j) \in A$
- $w_{ij} = \begin{cases} 1 & \text{if } (i,j) \text{ is installed} \\ 0 & \text{otherwise} \end{cases} \quad \forall (i,j) \in A$

$$z = \operatorname{Min} \sum_{(i,j) \in A} c_{ij} y_{ij} \qquad \text{(bandwidth cost)}$$

$$\sum_{p \in P_d} x_{dp} = h_d \quad \forall d \in D \qquad \text{(demand satisfaction)}$$

$$\sum_{d \in D} \sum_{p \in P_d} \delta_{ij}(dp)\, x_{dp} \leq y_{ij} \quad \forall (i,j) \in A \qquad \text{(capacity constraints)}$$

$$0 \leq y_{ij} \leq u_{ij} w_{ij} \quad \forall (i,j) \in A$$

$$\sum_{(i,j) \in A} K_{ij} w_{ij} \leq B \qquad \text{(budget constraint)}$$

$$x_{dp} \geq 0 \quad \forall d \in D, \forall p \in P_d$$

where

$$\delta_{ij}(dp) = \begin{cases} 1 & \text{if } (i,j) \in p,\ p \in P_d \\ 0 & \text{otherwise} \end{cases}$$

Obs : the problem extends the capacitated network design problem in (4.1.2)

- NP - Hard
- Optimization methods like Branch and Bound : ad hoc lower bound

Ad hoc lower bounding procedure

Assumption : for each (i,j) u_{ij} is an upper bound on the optimal capacity of (i,j) : therefore, the considered design problem with budget constraint indeed extends the uncapacitated network design problem in (4.1.1)

Key idea : for each fixed link installation, i.e. for each binary vector $\overline{w} = (\overline{w}_{ij})$, the bandwidth cost ($\equiv$ objective function) is minimized by applying the shortest path allocation rule (w.r.t. $\{c_{ij}\}$) in the subgraph $G(\overline{w})$ obtained by deleting all arcs (i,j) with $\overline{w}_{ij} = 0$.

- let $z(\overline{w})$ denote such a minimum cost for fixed $\overline{w}$, and $z(w)$ be the minimum cost for arbitrary w

Let $w^1 = (1, \dots 1)$, i.e. w^1 models the scenario where all links are installed, then:

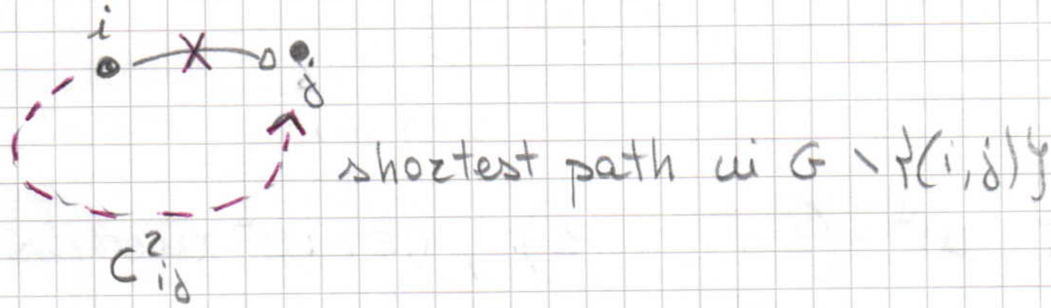
<u>Proposition 1</u>: $z(w) \geq z(w^1) \quad \forall w$.

<u>Proof</u>: $z(w^1)$ corresponds to eliminating (relaxing) the budget constraint $\square$

- <u>The lower bound $z(w^1)$</u> can therefore be computed via $|D|$ shortest path computations.

Can we enhance $z(w^1)$?

- for each $(i,j) \in A$, let C^1_{ij} denote the shortest path cost (w.r.t. $\{c_{ij}\}$) from i to j in G $(= G(w^1))$, while C^2_{ij} denote the shortest path cost from i to j in G minus (i,j):



- define

$$\eta_{ij} = h_{ij}\left(C^2_{ij} - C^1_{ij}\right) \qquad \forall (i,j) \in A,$$

where $h_{ij} = h_d$ if the pair (i,j) forms a commodity d, and 0 otherwise

<u>Hence</u>: η_{ij} is a local measure which accounts for the local effect of deleting (i,j) from G (note that $\eta_{ij} \geq 0$ since $C^2_{ij} \geq C^1_{ij}$ $\forall (i,j)$)

Proposition 2 :

$$z(w) \geq z(w^1) + \sum_{(i,j)\in A} (1 - w_{ij})\, \eta_{ij} \quad \forall w.$$

From Proposition 2, we can enhance the lower bound $z(w^1)$ by solving the following binary programming problem :

$$\text{Min } z(w^1) + \sum_{(i,j)\in A} (1 - w_{ij})\eta_{ij}$$

$$\sum_{(i,j)\in A} K_{ij}\, w_{ij} \leq B$$

$$w_{ij} \in \{0, 1\} \quad \forall (i,j)\in A$$

this can be obtained by solving :

$$\text{Max } \sum_{(i,j)\in A} \eta_{ij}\, w_{ij}$$

$$\sum_{(i,j)\in A} K_{ij}\, w_{ij} \leq B$$

$$w_{ij} \in \{0, 1\} \quad \forall (i,j)\in A$$

<u>0 - 1 Knapsack problem</u>

If U is the optimal value of the 0-1 Knapsack problem (or an upper bound, e.g. via its linear relaxation), then the enhanced lower bound is:

$$z(w^1) + \sum_{(i,j) \in A} \eta_{ij} - U$$

< The lower bounding approach can be generalized to scenarios where some links have been deleted (i.e. $w_{ij} = 0$) and others have been installed (i.e. $w_{ij} = 1$), as may happen in intermediate nodes of Branch and Bound tree for solving the stated design problem >

3 Shortest-path routing

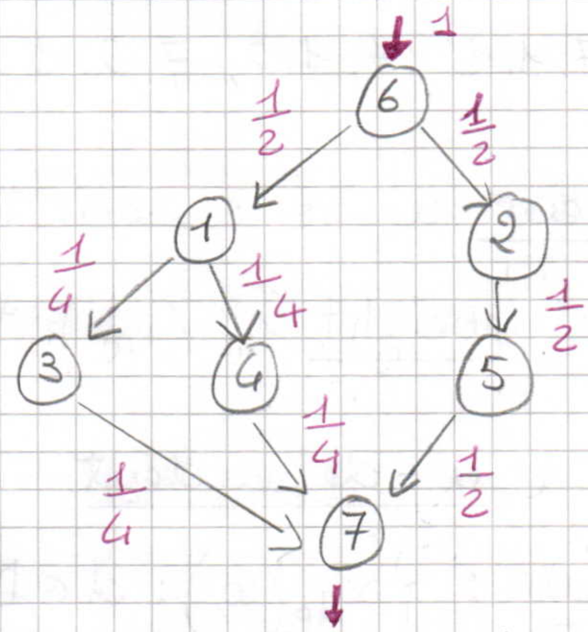
(Pióro - Medhi : in (7.1.1, 7.1.2, 7.2.1, 7.3.1))

- A link metric system w is associated with the links : w_{ij} weight of (i,j), $\forall (i,j) \in A$
- The metric induces a w-dependent multicommodity flow : $\{ x_{dp}(w) : d \in D, p \in P_d \}$
- Specifically : the request h_d of commodity d is sent along the shortest paths from the origin to the destination of d w.r.t $\{w_{ij}\}$

- This is typical of protocolos such as OSPF (Open Shortest Path First) in IP networks

- Usually, the flows are sent according to the ECMP rule, which splits the flow in "an equal way" among the shortest paths

example (ECMP rule)



routing template

(6, 7): origin-destination pair

assuming $w_{ij} = 1 \ \forall (i,j) \in A$ (hop-count metric), there are 3 shortest paths from 6 to 7: in red the shortest-path allocation rule according to ECMP

Observations

1) it is assumed $w_{ij} > 0 \ \forall (i,j) \in A$: therefore shortest simple paths ($\equiv$ no loops)

2) there is no simple function $x_{dp}(w)$ to express the flow along p for commodity d induced by w (ECMP rule)

< How to manage this ? >

(Mathematical?) formulations

Constraints

$$\sum_{p \in P_d} x_{dp}(w) = h_d \quad \forall d \in D$$

link utilization factor

$$\sum_{d \in D} \sum_{p \in P_d} \delta_{ij}(dp)\, x_{dp}(w) \le \gamma_{ij}\, u_{ij} \quad \forall (i,j) \in A$$

capacity of (i,j)

$w \in W$,

where a definition of W may be:

$$1 \le w_{ij} \le K \text{, \underline{integer}} \quad \forall (i,j) \in A$$

- For example, in the original specification of IS-IS protocol it is $K = 63$; such an upper bound has been then modified to $K = 2^{24} - 1$
- In OSPF it is $K = 2^{16} - 1$

Possible objective functions

- **Minimum average delay**

$$\text{Min} \sum_{(i,j)\in A} \frac{\bar{y}_{ij}}{(u_{ij} - \bar{y}_{ij})}$$

where $\bar{y}_{ij} = \sum_{d\in D} \sum_{p\in P_d} \delta_{ij}(dp) \cdot x_{dp}(w)$ is the load of link (i,j), $\forall (i,j)\in A$

* under certain assumptions, $\frac{1}{(u_{ij} - \bar{y}_{ij})}$ models the average packet delay on (i,j) (latency function of (i,j)) *

- **Minimization of maximum link utilization**

$$\text{Min } r$$

$$\sum_{d\in D} \sum_{p\in P_d} \delta_{ij}(dp)\, x_{dp}(w) \leq \gamma_{ij} \cdot u_{ij}\, r \quad \forall (i,j)\in A$$

replace the previous capacity constraints

ILP formulation

(Pioro - Médhi : 7.2.1)

Constraints

- Let (s, t) denote the generic origin-destination pair, i.e. $d \equiv (s, t)$ (so $h_d = h_{st}$)

- Replace variables $x_{dp}(w)$ by the following new sets of variables:

  r_{st} : length of the shortest path from s to t, $\forall$ pair (s, t)

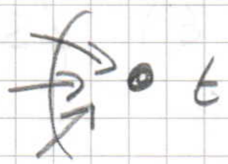
  $x_{(i,j)t}$: flow on (i,j) towards destination t, $\forall (i,j) \in A$, $t \in N \equiv$ node set

  y_{st} : common value of non-zero flow from s to t assigned to the links outgoing from s and belonging to the shortest paths from s to t

previous example $y_{67} = \frac{1}{2}$, $y_{17} = \frac{1}{4}$ ....

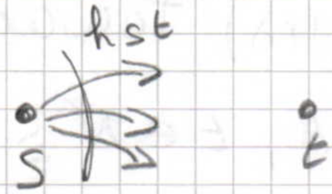
$$\alpha_{(i,j)t} = \begin{cases} 1 & \text{if } (i,j) \text{ belongs to a shortest path to } t \\ 0 & \text{otherwise} \end{cases} \qquad \forall (i,j) \in A,\ t \in N$$

1) $$\sum_{(i,t) \in BS(t)} x_{(i,t)t} = \sum_{s \neq t} h_{st} \qquad \forall t \in N$$



* the total flow entering t must be equal to the overall traffic demand having t as destination *

2) $$\sum_{(s,i) \in FS(i)} x_{(s,i)t} - \sum_{(i,s) \in BS(i)} x_{(i,s)t} = h_{st} \qquad \forall s, t \quad s \neq t$$



* the net flow leaving s and destined to t must be h_{st} *

3) $$\sum_{t \in N} x_{(i,j)t} \leq y_{ij} \cdot u_{ij} \; , \forall (i,j) \in A$$

* capacity constraints *

4) $$0 \le y_{it} - x_{(i,j)t} \le (1 - \alpha_{(i,j)t}) \sum_{s \neq t} h_{st}$$

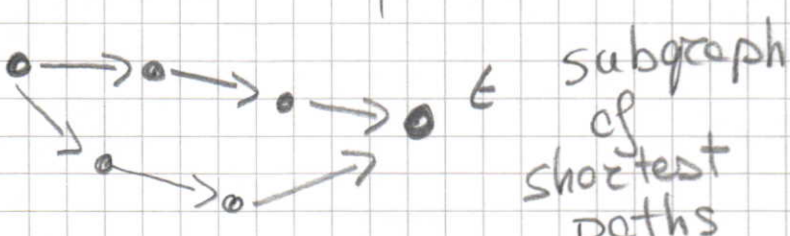
$$\forall t \in N, (i,j) \in A$$

* if (i,j) is on a shortest path to t (i.e. $\alpha_{(i,j)t} = 1$), then $x_{(i,j)t} = y_{it}$, i.e. the flow on (i,j) to t <u>must be equal</u> to the common value sent out from i to t (ECMP rule) *

5) $$x_{(i,j)t} \le \alpha_{(i,j)t} \sum_{s \neq t} h_{st} \qquad \forall (i,j) \in A, t \in N$$

* we can send flow on (i,j) to t only if (i,j) belongs to a shortest path to t *

<u>Essentially</u>: for each destination t, we are modelling the flow towards t (from the origins such that (s,t) is a commodity) along shortest paths towards t



We have now to impose that

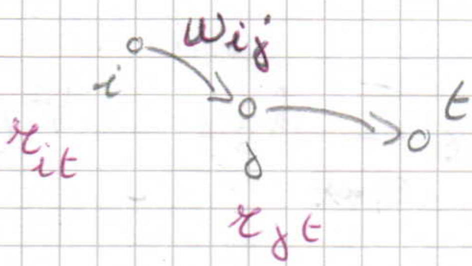
$\alpha_{(i,j)t} = 1$ if and only if (i,j) is on a shortest path towards t <u>w.r.t to $\{w_{ij}\}$</u>

6) $$0 \le r_{jt} + w_{ij} - r_{it} \le (1 - \alpha_{(ij)t})\, \underset{\text{large number}}{M}$$

$$\forall (i,j) \in A,\ t \in N$$

* if $\alpha_{(i,j)t} = 1$, then

$$w_{ij} + r_{jt} = r_{it}$$



$\forall t \in N$

These are Bellman's shortest path optimality conditions! (reversed since they address shortest paths <u>to</u> a given destination) *

Finally, we need to impose that, if (i,j) does not belong to a shortest path to t, i.e. $w_{ij} + r_{jt} > r_{it}$, then (i,j) is not used for t, i.e. $\alpha_{(i,j)t} = 0$:

7) $$w_{ij} + r_{jt} - r_{it} \geq 1 - \alpha_{(i,j)t} \quad \forall (i,j) \in A, \; t \in N$$

8) $$w_{ij} \geq 1 \quad \forall (i,j) \in A$$

* choice of the link metric system *

Note that 7) is corrected ($w_{ij} + r_{jt} - r_{it} > 0 \Rightarrow w_{ij} + r_{jt} - r_{it} \geq 1$) according to constraints 8).

<u>If we would add</u>:

$$\sum_{(s,i) \in FS(s)} \alpha_{(si)t} \overset{(\leq)}{=} 1 \quad \forall s, t, \; s \neq t$$

then we would force a unique shortest path (w.r.t $\{w_{ij}\}$) for each pair (s,t)

- "difficult" ILP formulation (also for moderate size networks)

- anyway, it allows exact approaches (ILP solver, Branch and Bound...) for small size networks

- alternative approaches : heuristics

<u>Weight Adjustement (WA) heuristic</u>

<u>Obs</u> : for each fixed $\{w_{ij}\}$, the shortest-path routing is easy : just implements the ECMP rule.

<u>So</u> : how can we modify $\{w_{ij}\}$ by imposing (for example)

$$1 \leq w_{ij} \leq K \text{ , integer } \quad \forall (i,j) \in A \text{ ?}$$

- (WA) starts with randomly generated $\{w_{ij}\}$, $1 \leq w_{ij} \leq K$ integer $\forall (i,j) \in A$.

- Then (WA) iterates based on two local search procedures:

  - weight adjustement:

    - if the "load" of (i,j) is greater than $\gamma_{ij} u_{ij}$ for some (i,j) (overloaded links), then "increase" w_{ij} to attempt to remove some flow from (i,j)

    - if (i,j) is under-loaded, then w_{ij} "may" be decreased

    < strive toward feasibility >

  - load optimization:

    if all links are under-loaded (= feasible routing), then select two links (e.g. most loaded and least loaded), adjuste their weights (e.g. + 1 and - 1), depending on the goal of the optimization (objective function)

* no further details *

4 Restoration and protection design in resilient networks

- Network optimization problems for a selected ~~set~~ of failure scenarios (rather than normal operating state)

- The resulting optimization problems are referred to as restoration design problems: design networks which are resilient (robust) to failures, i.e. are able to carry (possible reduced) traffic demands also when part of network resources fail