The Load Balancing Optimization of Telecommunication Networks

Dorabella Santos[1] Amaro de Sousa[1,2] Filipe Alvelos[3]

[1]Instituto de Telecomunicações - Pólo de Aveiro
Portugal

[2]Departamento de Electrónica, Telecomunicações e Informática
Universidade de Aveiro
Portugal

[3]Centro Algoritmi/Departmento de Produção e Sistemas
Universidade do Minho
Portugal

Introduction

Single path routing

- Capacitated telecommunications network

Introduction

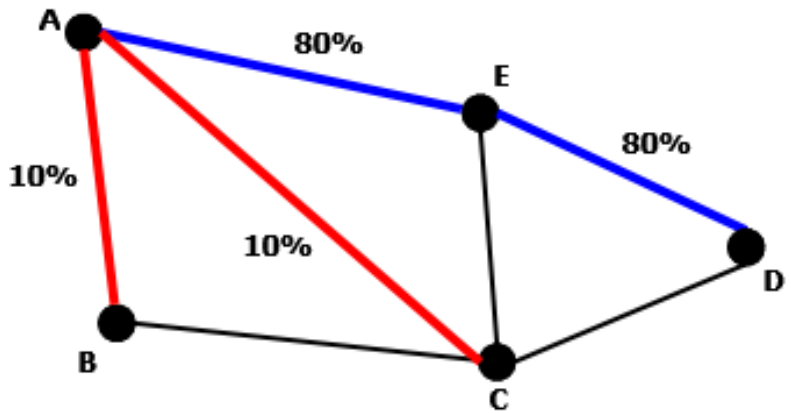
Single path routing

- Capacitated telecommunications network
- Traffic commodities are to be supported by the network
  - single path routing

Introduction

Single path routing

- Capacitated telecommunications network
- Traffic commodities are to be supported by the network
  - single path routing
- Example:
  - Links have capacity 100 Mbps
  - Commodity A-D has demand 80 Mbps
  - Commodity B-C has demand 10 Mbps



Introduction

Network load balancing

- Aim: commodities are to be routed such that network link load is optimized

Introduction

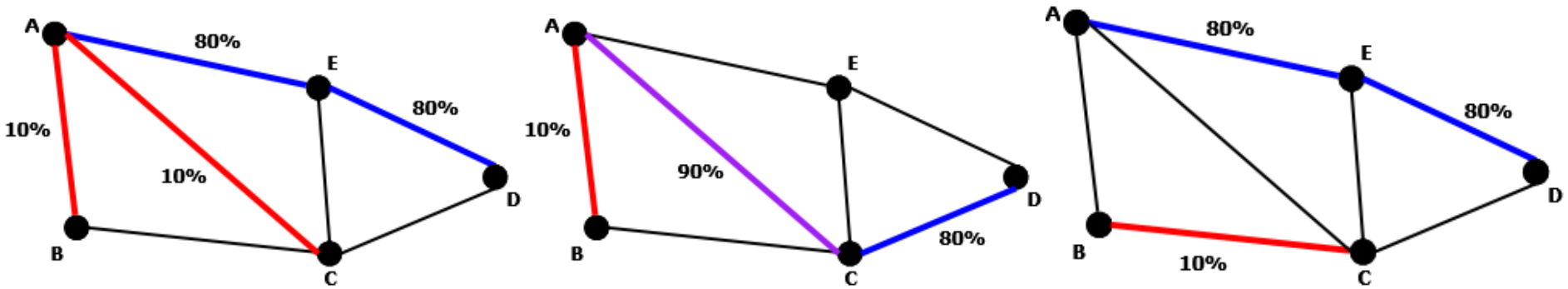
Network load balancing

- Aim: commodities are to be routed such that network link load is optimized
  - minimize the worst link load
  - minimize the second worst link load, guaranteeing that the worst link load is optimized
  - minimize the third link load, guaranteeing that the two worst link loads are optimized
  - ...

Introduction

Network load balancing

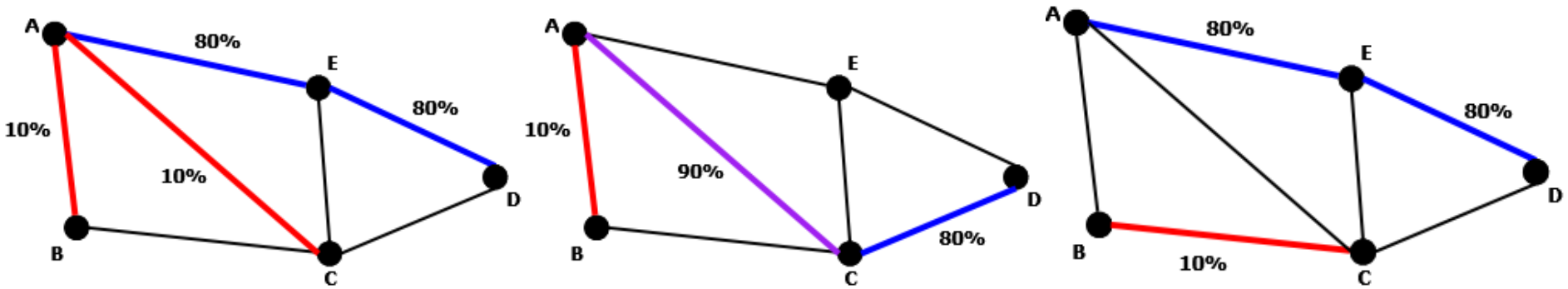
- Aim: commodities are to be routed such that network link load is optimized
  - minimize the worst link load
  - minimize the second worst link load, guaranteeing that the worst link load is optimized
  - minimize the third link load, guaranteeing that the two worst link loads are optimized
  - ...



Introduction

Network load balancing

- Aim: commodities are to be routed such that network link load is optimized
  - minimize the worst link load
  - minimize the second worst link load, guaranteeing that the worst link load is optimized
  - minimize the third link load, guaranteeing that the two worst link loads are optimized
  - ...



↪ Network robustness to unpredictable traffic demand growth

Parameters

- $G(N, A)$
- $c_{\{ij\}}$ capacity of edge $\{i, j\} \in A$
- K set of commodities
  - each commodity $k \in K$ is characterized by its origin node o_k, its destination node d_k and its demand $b_k > 0$
- P_k set of candidate paths for commodity $k \in K$
- $\delta^{pk}_{\{ij\}}$ binary parameter
  - indicates if edge $\{i, j\}$ is in path $p \in P_k$

Decision variables

- $\mu_{\{ij\}} \in [0,1]$ – load in edge $\{i,j\} \in A$
- $\varphi_k^p \in \{0,1\}$ – indicates if candidate $p \in P_k$ is chosen as the path for routing $k \in K$

Network Load Balancing Optimization

Mathematical formulation

Decision variables

- $\mu_{\{ij\}} \in [0,1]$ – load in edge $\{i,j\} \in A$
- $\varphi_k^p \in \{0,1\}$ – indicates if candidate $p \in P_k$ is chosen as the path for routing $k \in K$

Constraints

$$\sum_{p \in P_k} \varphi_k^p = 1 \qquad \forall k \in K \tag{1}$$

$$\sum_{k \in K} \sum_{p \in P_k} b_k \delta_{\{ij\}}^{pk} \varphi_k^p = c_{\{ij\}} \mu_{\{ij\}} \qquad \forall \{i,j\} \in A \tag{2}$$

$$\mu_{\{ij\}} \in [0,1],\ \varphi_k^p \in \{0,1\} \tag{3}$$

Network Load Balancing Optimization

Mathematical formulation

Optimization Objective

- Minimize worst link load; minimize second worst link load without jeopardizing the worst link load; minimize third worst link load without jeopardizing the two worst link loads; ...

Optimization Objective

- Minimize worst link load; minimize second worst link load without jeopardizing the worst link load; minimize third worst link load without jeopardizing the two worst link loads; ...

  $\boldsymbol{\mu} = (\mu_{\{ij\}} : \{i,j\} \in A)$

Optimization Objective

- Minimize worst link load; minimize second worst link load without jeopardizing the worst link load; minimize third worst link load without jeopardizing the two worst link loads; ...

  $\boldsymbol{\mu} = (\mu_{\{ij\}} : \{i,j\} \in A)$

  $[\boldsymbol{\mu}]$ is the vector obtained from $\boldsymbol{\mu}$ by rearranging its elements in non-increasing order $\Rightarrow$ $[\boldsymbol{\mu}]_l$ is the l^{th} element of $[\boldsymbol{\mu}]$

Optimization Objective

- Minimize worst link load; minimize second worst link load without jeopardizing the worst link load; minimize third worst link load without jeopardizing the two worst link loads; ...

  $\boldsymbol{\mu} = (\mu_{\{ij\}} : \{i,j\} \in A)$

  $[\boldsymbol{\mu}]$ is the vector obtained from $\boldsymbol{\mu}$ by rearranging its elements in non-increasing order $\Rightarrow$ $[\boldsymbol{\mu}]_l$ is the l^{th} element of $[\boldsymbol{\mu}]$

  lexmin $[\mu]$

  s.t.

  $(1) - (3)$

Optimization Objective

- Minimize worst link load; minimize second worst link load without jeopardizing the worst link load; minimize third worst link load without jeopardizing the two worst link loads; ...

  $\boldsymbol{\mu} = (\mu_{\{ij\}} : \{i,j\} \in A)$

  $[\boldsymbol{\mu}]$ is the vector obtained from $\boldsymbol{\mu}$ by rearranging its elements in non-increasing order $\Rightarrow$ $[\boldsymbol{\mu}]_l$ is the l^{th} element of $[\boldsymbol{\mu}]$

  lexmin $[\mu]$

  s.t.

  $(1) - (3)$

↪ Vector $\mathbf{a} = (a_1, ..., a_m)$ is lexicographically smaller than $\mathbf{b} = (b_1, ..., b_m)$ if $a_1 < b_1$ or if there exists $k \in \{1, ..., m-1\}$ such that $a_i = b_i$ for $i \leq k$ and $a_{k+1} < b_{k+1}$

$(1, 3, 2)$ is lexicographically smaller than $(1, 5, 1)$

Network Load Balancing Optimization

Solution approach

- Non-linear problem $\Rightarrow$ can be solved as a sequence of MILPs

Network Load Balancing Optimization

Solution approach

- Non-linear problem $\Rightarrow$ can be solved as a sequence of MILPs
- Computationally hard to solve

Network Load Balancing Optimization

Solution approach

- Non-linear problem $\Rightarrow$ can be solved as a sequence of MILPs
- Computationally hard to solve

  $\Downarrow$

- Hybridization of column generation and metaheuristics

Network Load Balancing Optimization

Solution approach

- Non-linear problem $\Rightarrow$ can be solved as a sequence of MILPs
- Computationally hard to solve

$\Downarrow$

- Hybridization of column generation and metaheuristics

$\hookrightarrow$ Hybrid CG and GRASP+PR

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity
  - neighborhood of a solution: set of solutions which differ from the current one on a single path

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity
  - neighborhood of a solution: set of solutions which differ from the current one on a single path
  - initial solutions: greedy randomized

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity
  - neighborhood of a solution: set of solutions which differ from the current one on a single path
  - initial solutions: greedy randomized
  - path relinking (PR) between local search procedures and elite list

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity
  - neighborhood of a solution: set of solutions which differ from the current one on a single path
  - initial solutions: greedy randomized
  - path relinking (PR) between local search procedures and elite list
  - elite list has distance: elite members must be different for a minimum number of paths

Hybrid CG and GRASP+PR

Heuristic proposal

- LP column generation (CG) for the load balancing problem
  - set of paths for each commodity $k \in K$
- Multi-start local search heuristics (GRASP) over the pool of CG paths
  - solution: set of one path per commodity
  - neighborhood of a solution: set of solutions which differ from the current one on a single path
  - initial solutions: greedy randomized
  - path relinking (PR) between local search procedures and elite list
  - elite list has distance: elite members must be different for a minimum number of paths

$\hookrightarrow$ Search space constrained to paths obtained by CG

Hybrid CG and GRASP+PR

Heuristic proposal

- P = linear relaxation of the load balancing problem
- Ω = search space

Hybrid CG and GRASP+PR

Heuristic proposal

- P = linear relaxation of the load balancing problem
- Ω = search space

1. $\Phi \leftarrow \text{CG}(P)$
2. $\Omega \leftarrow \Phi$
3. **while** time $<$ *MaxTime* **do**
4. $\quad [\mu] \leftarrow \text{GRASP+PR}(\Omega)$
5. **end**

Hybrid CG and GRASP+PR

Heuristic proposal

- P = linear relaxation of the load balancing problem
- Ω = search space

1. $\Phi \leftarrow \text{CG}(P)$
2. $\Omega \leftarrow \Phi$
3. **while** time $<$ *MaxTime* **do**
4. $\quad [\mu] \leftarrow \text{GRASP+PR}(\Omega)$
5. **end**

$\hookrightarrow$ The solution which provides $[\mu]$ is a good quality solution provided that Φ has good quality paths

Hybrid CG and GRASP+PR

Heuristic proposal

- P = linear relaxation of the load balancing problem
- Ω = search space

1. $\Phi \leftarrow \text{CG}(P)$
2. $\Omega \leftarrow \Phi$
3. **while** time $<$ *MaxTime* **do**
4. $\quad [\mu] \leftarrow \text{GRASP+PR}(\Omega)$
5. **end**

↪ The solution which provides $[\mu]$ is a good quality solution provided that Φ has good quality paths

↪ If Φ does not have good quality paths $\Rightarrow$ add new paths to the search space

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$
  - $[\eta]$ is lexicographically smaller or equal to $[\mu]$

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$
  - $[\eta]$ is lexicographically smaller or equal to $[\mu]$
- Solve P' using CG

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$
  - $[\eta]$ is lexicographically smaller or equal to $[\mu]$
- Solve P' using CG
- If new paths are generated, add to search space

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$
  - $[\eta]$ is lexicographically smaller or equal to $[\mu]$
- Solve P' using CG
- If new paths are generated, add to search space

1. $(\Phi, [\eta]) \leftarrow \text{CG}(P, \{\})$
2. $\Omega \leftarrow \Phi$
3. **while** time $<$ *MaxTime* **do**
4. $\quad [\mu] \leftarrow \text{GRASP+PR}(\Omega, \textit{Criteria})$
5. $\quad P' \leftarrow \text{Perturbation}([\mu], [\eta])$
6. $\quad \Phi \leftarrow \text{CG}(P', \Omega)$
7. $\quad \Omega \leftarrow \Omega \cup \Phi$
8. **end**

Hybrid CG and GRASP+PR

Heuristic proposal: add new paths

- Define a perturbed problem P' based on $[\mu]$ and LP solution loads $[\eta]$
  - $[\eta]$ is lexicographically smaller or equal to $[\mu]$
- Solve P' using CG
- If new paths are generated, add to search space

1. $(\Phi, [\eta]) \leftarrow \text{CG}(P, \{\})$
2. $\Omega \leftarrow \Phi$
3. **while** time $<$ *MaxTime* **do**
4. $\quad [\mu] \leftarrow \text{GRASP+PR}(\Omega, \textit{Criteria})$
5. $\quad P' \leftarrow \text{Perturbation}([\mu], [\eta])$
6. $\quad \Phi \leftarrow \text{CG}(P', \Omega)$
7. $\quad \Omega \leftarrow \Omega \cup \Phi$
8. **end**

$\hookrightarrow$ If the search space grows too much, the search can be jeopardized $\Rightarrow$ remove some paths

Hybrid CG and GRASP+PR

Heuristic proposal: add and remove paths

- Compute number of times each path is used by the local optimum solutions

Hybrid CG and GRASP+PR

Heuristic proposal: add and remove paths

- Compute number of times each path is used by the local optimum solutions
- Columns that are never used, E, can be excluded from Ω

Hybrid CG and GRASP+PR

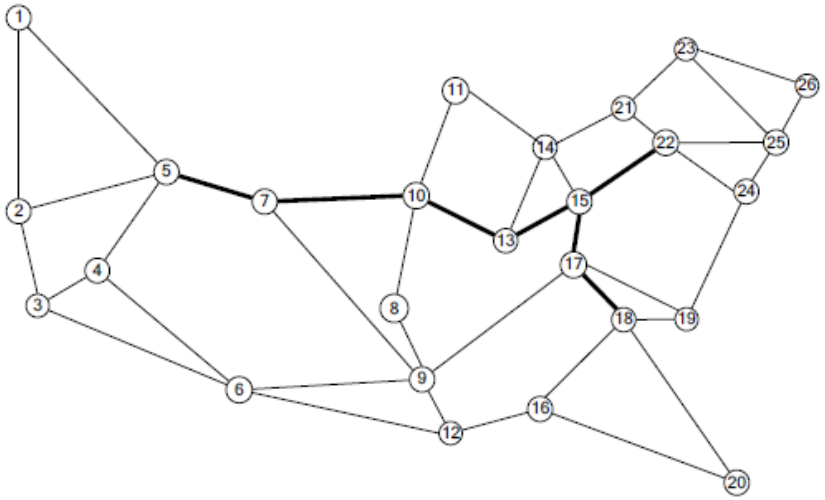
Heuristic proposal: add and remove paths

- Compute number of times each path is used by the local optimum solutions
- Columns that are never used, E, can be excluded from Ω

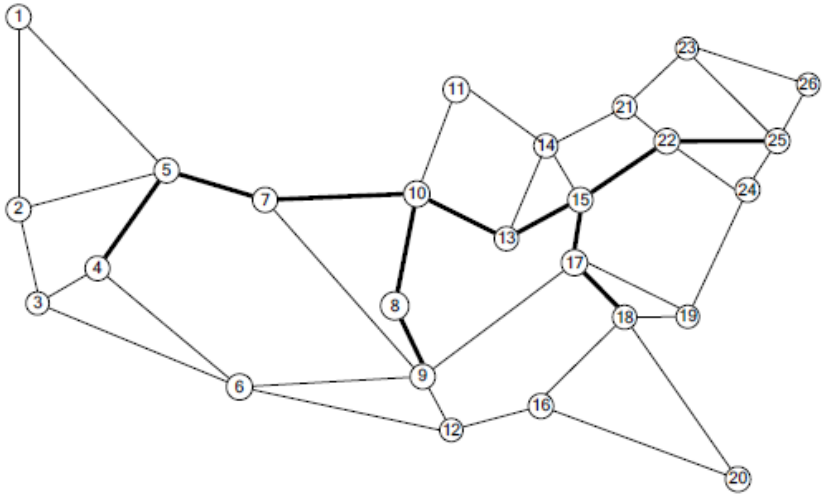
1. $(\Phi, [\eta]) \leftarrow \text{CG}(P, \{\})$
2. $\Omega \leftarrow \Phi$
3. $\Theta \leftarrow \Phi$
4. **while** time $<$ *MaxTime* **do**
5. $\quad([\mu], E) \leftarrow \text{GRASP+PR}(\Omega, \textit{Criteria})$
6. $\quad P' \leftarrow \text{Perturbation}([\mu], [\eta])$
7. $\quad \Phi \leftarrow \text{CG}(P', \Theta)$
8. $\quad \Theta \leftarrow \Theta \cup \Phi$
9. $\quad \Omega \leftarrow \Omega \cup \Phi \cap E$
10. **end**

Computational Results

NSF network configurations



NSF'



NSF''

1 Gbps; 10 Gbps in bold

Computational Results

Instances

- NSF network
  - 26 nodes; 42 links

Computational Results

Instances

- NSF network
  - 26 nodes; 42 links
- 24 instances:
  - 12 NSF' instances; 12 NSF'' instances

Computational Results

Instances

- NSF network
  - 26 nodes; 42 links
- 24 instances:
  - 12 NSF' instances; 12 NSF'' instances
- For each instance a demand matrix was randomly generated
  - emulate different possible real scenarios

Computational Results

Instances

- NSF network
  - 26 nodes; 42 links
- 24 instances:
  - 12 NSF' instances; 12 NSF'' instances
- For each instance a demand matrix was randomly generated
  - emulate different possible real scenarios
- Half of each group have 325 commodities (H) and half have 190 commodities (L)

Computational Results

Instances

- NSF network
  - 26 nodes; 42 links
- 24 instances:
  - 12 NSF' instances; 12 NSF" instances
- For each instance a demand matrix was randomly generated
  - emulate different possible real scenarios
- Half of each group have 325 commodities (H) and half have 190 commodities (L)
- Applying CG to the linear relaxation of the load balancing problem:
  - for each instance, the number of paths generated on average per commodity is ≈ 4

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP
- Still computationally demanding

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP
- Still computationally demanding
- CPLEX 12.1; solve 8 worst link loads within timelimit = 6 hours

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP
- Still computationally demanding
- CPLEX 12.1; solve 8 worst link loads within timelimit = 6 hours
- Group G:
  - 4 NSF' instances and 6 NSF" instances
  - half of these instances are of group H
  - at least 3 worst loads were solved and are optimal
  - integer RMP values coincide with LP values

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP
- Still computationally demanding
- CPLEX 12.1; solve 8 worst link loads within timelimit = 6 hours
- Group G:
  - 4 NSF' instances and 6 NSF" instances
  - half of these instances are of group H
  - at least 3 worst loads were solved and are optimal
  - integer RMP values coincide with LP values
- Group B:
  - all the other cases
  - the worst load value does not coincide with the LP value
  - for 2 NSF' instances and 2 NSF" instances the worst load was not even solved within the timelimit
  - may be optimal or not

Computational Results

Integer RMP

- Solve load balancing problem only with the paths given by the LP
- Still computationally demanding
- CPLEX 12.1; solve 8 worst link loads within timelimit = 6 hours
- Group G:
  - 4 NSF' instances and 6 NSF" instances
  - half of these instances are of group H
  - at least 3 worst loads were solved and are optimal
  - integer RMP values coincide with LP values
- Group B:
  - all the other cases
  - the worst load value does not coincide with the LP value
  - for 2 NSF' instances and 2 NSF" instances the worst load was not even solved within the timelimit
  - may be optimal or not

↪ The LP paths contain near optimal solutions for group G

Computational Results

Hybrid CG and GRASP+PR

Search space is not modified (no paths are added):

- Results between PR with incumbent (elite size = 1) and PR with elite list

Computational Results

Hybrid CG and GRASP+PR

Search space is not modified (no paths are added):

- Results between PR with incumbent (elite size = 1) and PR with elite list
- Elite list: size = 20; distance = 30%

Computational Results

Hybrid CG and GRASP+PR

Search space is not modified (no paths are added):

- Results between PR with incumbent (elite size = 1) and PR with elite list
- Elite list: size = 20; distance = 30%
- Timelimit = 60 seconds; 10 runs for each instance and PR case

Computational Results

Hybrid CG and GRASP+PR

Search space is not modified (no paths are added):

- Results between PR with incumbent (elite size = 1) and PR with elite list
- Elite list: size = 20; distance = 30%
- Timelimit = 60 seconds; 10 runs for each instance and PR case
- PR with incumbent:
  - for instances of group G the solutions are of good quality ⇒ almost always the 3 worst link loads are optimal

Computational Results

Hybrid CG and GRASP+PR

- PR with elite list:

Instance Sets	Improvement
Global	38.1%
B	56.4%
G	12.8%
H	56.4%
L	17.4%

Computational Results

Hybrid CG and GRASP+PR

- PR with elite list:

Instance Sets	Improvement
Global	38.1%
B	56.4%
G	12.8%
H	56.4%
L	17.4%

↪ Overall improvement using elite list

↪ Improvement is more significant for groups B and H

↪ Average occupancy of elite list is between 4 and 5 $\Rightarrow$ PR does not penalize performance

Computational Results

Hybrid CG and GRASP+PR

Search space is modified (new paths added and/or some paths removed):

- PR with elite list: size $= 20$; distance $= 30\%$

Computational Results

Hybrid CG and GRASP+PR

Search space is modified (new paths added and/or some paths removed):

- PR with elite list: size $= 20$; distance $= 30\%$
- Perturbed problem P_1 aims at lowering worst link load of previous incumbent that does not coincide with LP value
- Perturbed problem P_2 assumes that the worst link load of the incumbent that differs from the LP value is optimal, and aims at lowering the next worst link load

Computational Results

Hybrid CG and GRASP+PR

Search space is modified (new paths added and/or some paths removed):

- PR with elite list: size = 20; distance = 30%
- Perturbed problem P_1 aims at lowering worst link load of previous incumbent that does not coincide with LP value
- Perturbed problem P_2 assumes that the worst link load of the incumbent that differs from the LP value is optimal, and aims at lowering the next worst link load
- Timelimit = 60 seconds; 10 runs for each instance and case

Computational Results

Hybrid CG and GRASP+PR

	P_1 without exclusion of paths		P_1 with exclusion of paths	
Sets	Modifications	Improvement	Modifications	Improvement
Global	7.2	63.2%	7.1	70.3%
B	6.3	56.4%	6.4	56.4%
G	8.6	77.8%	8.0	85.7%
H	6.7	43.9%	6.2	56.4%
L	7.8	85.7%	7.9	85.7%

Computational Results

Hybrid CG and GRASP+PR

	P_1 without exclusion of paths		P_1 with exclusion of paths	
Sets	Modifications	Improvement	Modifications	Improvement
Global	7.2	63.2%	7.1	70.3%
B	6.3	56.4%	6.4	56.4%
G	8.6	77.8%	8.0	85.7%
H	6.7	43.9%	6.2	56.4%
L	7.8	85.7%	7.9	85.7%

- ↪ Number of added paths does not increase significantly the size of the restricted search space
- ↪ Number of removed paths is of the same order of magnitude of the number of added paths

Computational Results

Hybrid CG and GRASP+PR

	P_1 without exclusion of paths		P_1 with exclusion of paths	
Sets	Modifications	Improvement	Modifications	Improvement
Global	7.2	63.2%	7.1	70.3%
B	6.3	56.4%	6.4	56.4%
G	8.6	77.8%	8.0	85.7%
H	6.7	43.9%	6.2	56.4%
L	7.8	85.7%	7.9	85.7%

- ↪ Number of added paths does not increase significantly the size of the restricted search space
- ↪ Number of removed paths is of the same order of magnitude of the number of added paths
- ↪ Adding paths using P_1 shows improvement; slightly better performance when exclusion of paths is used
- ↪ Significant improvements for groups G and L

Computational Results

Hybrid CG and GRASP+PR

P_2:

- ↪ Number of modifications of P_2 similar to P_1
- ↪ P_2 adds much less paths and excludes much more paths $\Rightarrow$ the restricted search space decreases in size
- ↪ P_2 shows slight improvement when exclusion of paths is used
- ↪ P_2 performs worse than P_1

Computational Results

Hybrid CG and GRASP+PR

P_2:

- $\hookrightarrow$ Number of modifications of P_2 similar to P_1
- $\hookrightarrow$ P_2 adds much less paths and excludes much more paths $\Rightarrow$ the restricted search space decreases in size
- $\hookrightarrow$ P_2 shows slight improvement when exclusion of paths is used
- $\hookrightarrow$ P_2 performs worse than P_1

$\Downarrow$

P_1/P_2 combined version

Computational Results

Hybrid CG and GRASP+PR

	P_1/P_2 without exclusion of paths		P_1/P_2 with exclusion of paths	
Sets	Modifications	Improvement	Modifications	Improvement
Global	4.9	56.4%	5.0	94.1%
B	4.4	56.4%	4.4	112,5%
G	5.6	50.0%	5.8	70.3%
H	4.8	85.7%	4.8	112.5%
L	5.2	32.6%	5.1	77.8%

Computational Results

Hybrid CG and GRASP+PR

	P_1/P_2 without exclusion of paths		P_1/P_2 with exclusion of paths	
Sets	Modifications	Improvement	Modifications	Improvement
Global	4.9	56.4%	5.0	94.1%
B	4.4	56.4%	4.4	112,5%
G	5.6	50.0%	5.8	70.3%
H	4.8	85.7%	4.8	112.5%
L	5.2	32.6%	5.1	77.8%

- ↪ Significant decrease in the number of modifications of the search space
- ↪ Number of added paths similar to P_1
- ↪ Number of excluded paths significantly lower than P_1
- ↪ Combined performance with path exclusion is the best

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)
- PR shows improvement when using elite list with a distance large enough $\Rightarrow$ average occupancy of list should be low
  - favors especially instances from groups B and H

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)
- PR shows improvement when using elite list with a distance large enough $\Rightarrow$ average occupancy of list should be low
  - favors especially instances from groups B and H
- To improve furthermore: add new paths using P_1 and/or P_2

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)
- PR shows improvement when using elite list with a distance large enough $\Rightarrow$ average occupancy of list should be low
  - favors especially instances from groups B and H
- To improve furthermore: add new paths using P_1 and/or P_2
  - P_1 results in overall better solutions especially when some paths are removed
    - overall improvement of 70.3%
    - favors especially instances of groups G and L

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)
- PR shows improvement when using elite list with a distance large enough $\Rightarrow$ average occupancy of list should be low
  - favors especially instances from groups B and H
- To improve furthermore: add new paths using P_1 and/or P_2
  - P_1 results in overall better solutions especially when some paths are removed
    - overall improvement of 70.3%
    - favors especially instances of groups G and L
  - P_2 has poor overall performance

Conclusions

Hybrid CG and GRASP+PR applied to network load balancing

- GRASP+PR performed on restricted search space $\Rightarrow$ search is more efficient
- Heuristic solutions are of good quality for instances of group G (PR with incumbent)
- PR shows improvement when using elite list with a distance large enough $\Rightarrow$ average occupancy of list should be low
  - favors especially instances from groups B and H
- To improve furthermore: add new paths using P_1 and/or P_2
  - P_1 results in overall better solutions especially when some paths are removed
    - overall improvement of 70.3%
    - favors especially instances of groups G and L
  - P_2 has poor overall performance
  - P_1/P_2 combined version has the best overall performance especially when some paths are removed
    - overall improvement of 94.1%
    - favors significantly all groups especially B and H

Conclusions

Hybrid CG and GRASP+PR: future work

- Apply heuristic proposal to more complex load balancing problems:
  - Load balancing using multiple spanning trees
    - ↪ D. Santos, A. de Sousa, F. Alvelos, M. Pióro, Optimization of Network Load Balancing in Multiple Spanning Tree Routing Networks, Telecommunication Systems Journal, 48(1-2), pp 109-124, Springer, 2011
  - Load balancing with path protection
    - ↪ A. de Sousa, D. Santos, P. Matos and J. Madeira, Load balancing optimization of capacitated networks with path protection, Electronic Notes, 36, pp. 1249-1256, 2010

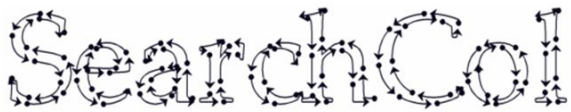
Conclusions

Hybrid CG and GRASP+PR: SearchCol

- Complementary work to

↪ D. Santos, A. de Sousa, F. Alvelos, M. Pióro, Link Load Balancing Optimization of Telecommunication Networks: a Column Generation based Heuristic Approach, NETWORKS 2010, Warsaw, Poland, September, 2010

↪ F. Alvelos, A. de Sousa, D. Santos, SearchCol: Metaheuristic Search by Column Generation, 7th International Workshop on Hybrid Metaheuristics, Vienna, Austria, October, 2010



Metaheuristic Search by Column Generation

http://searchcol.dps.uminho.pt