Algorithmic Graph Theory: How hard is your combinatorial optimization problem?

Arie M.C.A. Koster

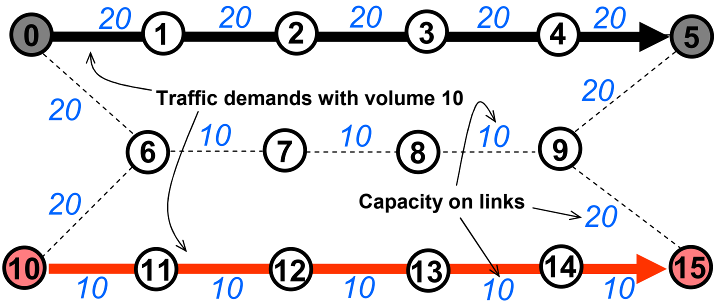
Lecture 2

Clemson, June 7, 2017

CLEMSON UNIVERSITY
Lehrstuhl II für Mathematik
RWTH AACHEN UNIVERSITY

Network Design with Compression:

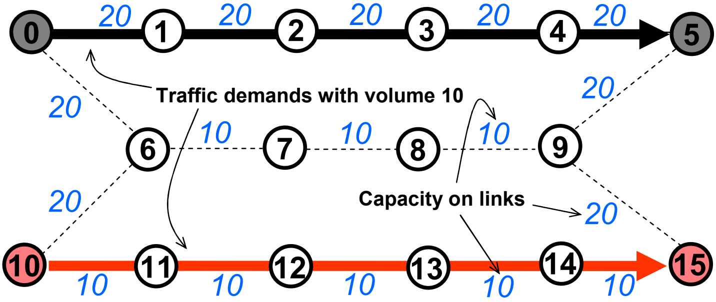
- Given a Network $G = (V, E)$,
- With capacity $c_{uv} = c \geq 0$ for all edges uv.
- 2 Demands d^1, d^2 (with a potential compression rate λ).



Figures provided by Truong Khoa Phan

Network Design with Compression:

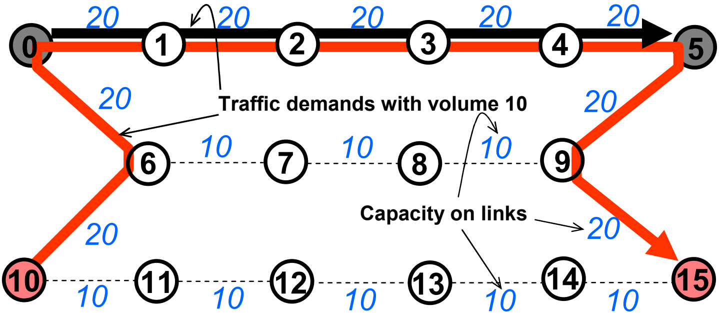
- Given a Network $G = (V, E)$,
- With capacity $c_{uv} = c \geq 0$ for all edges uv.
- 2 Demands d^1, d^2 (with a potential compression rate λ).
- Find a feasible routing



Figures provided by Truong Khoa Phan

Network Design with Compression:

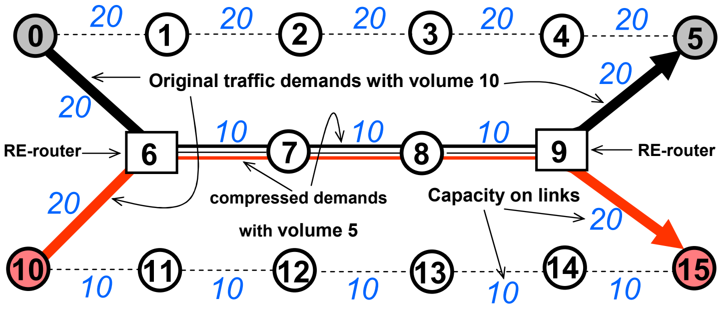
- Given a Network $G = (V, E)$,
- With capacity $c_{uv} = c \geq 0$ for all edges uv.
- 2 Demands d^1, d^2 (with a potential compression rate λ).
- Find a feasible routing with minimal energy costs.



Figures provided by Truong Khoa Phan

Network Design with Compression:

- Given a Network $G = (V, E)$,
- With capacity $c_{uv} = c \geq 0$ for all edges uv.
- 2 Demands d^1, d^2 (with a potential compression rate λ).
- Find a feasible routing with minimal energy costs.
- Employ Compression if beneficial.



Figures provided by Truong Khoa Phan

- Network Design with Compression can be fomulated as mixed integer program.

- Network Design with Compression can be fomulated as mixed integer program.
- Set of commodities Q and capacity c installable in integers.

NDPC as MILP

- Network Design with Compression can be fomulated as mixed integer program.
- Set of commodities Q and capacity c installable in integers.
- Combine flow-conservation, capacity restrictions and Compression.

- NETWORK DESIGN WITH COMPRESSION can be fomulated as mixed integer program.
- Set of commodities Q and capacity c installable in integers.
- Combine flow-conservation, capacity restrictions and Compression.
- Find a feasible routing,
  Minimizing energy consumption (C_{uv} for $uv \in E$ and C_v for active compression at $v \in V$).

- NETWORK DESIGN WITH COMPRESSION can be fomulated as mixed integer program.
- Set of commodities Q and capacity c installable in integers.
- Combine flow-conservation, capacity restrictions and Compression.
- Find a feasible routing,
  Minimizing energy consumption (C_{uv} for $uv \in E$ and C_v for active compression at $v \in V$).
- Variables:

$f_{vu}^{st} \in \mathbb{R}_{\geq 0}$:	Fraction of demand st routed uncompressed on edge vu.
$g_{vu}^{st} \in \mathbb{R}_{\geq 0}$:	Fraction of demand st routed compressed on edge vu.
$x_{uv} \in \mathbb{Z}_{\geq 0}$:	Usage of edge uv.
$y_v \in \{0, 1\}$:	Whether compression enabled at node v.

NDPC as MILP

$$\min \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v$$

$$\text{s.t.} \sum_{u \in N(v)} \left(f^q_{vu} + g^q_{vu} - f^q_{uv} - g^q_{uv}\right) = \begin{cases} -1 & \text{if } u = s^q, \\ 1 & \text{if } u = t^q, \\ 0 & \text{else} \end{cases} \qquad \forall\, v \in V, \forall\, q \in Q$$

$$\sum_{q \in Q} \left(d^q \left(f^q_{uv} + f^q_{vu}\right) + \lambda d^q \left(g^q_{uv} + g^q_{vu}\right)\right) \leq c x_{uv} \qquad \forall\, uv \in E$$

$$-y_v \leq \sum_{u \in N(v)} \left(g^q_{uv} - g^q_{vu}\right) \leq y_v \qquad \forall\, v \in V, \forall\, q \in Q$$

$$x_{uv} \in \mathbb{Z}_{\geq 0}, y_v \in \{0,1\}, f^q_{uv} \geq 0, g^q_{uv} \geq 0$$

NDPC as MILP

$$
\begin{aligned}
\min \ & \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v \\
\text{s.t.} \ & \sum_{u \in N(v)} \left(f^q_{vu} + g^q_{vu} - f^q_{uv} - g^q_{uv}\right) = \begin{cases} -1 & \text{if } u = s^q, \\ 1 & \text{if } u = t^q, \\ 0 & \text{else} \end{cases} && \forall\, v \in V, \forall\, q \in Q \\
& \sum_{q \in Q} \left(d^q \left(f^q_{uv} + f^q_{vu}\right) + \lambda d^q \left(g^q_{uv} + g^q_{vu}\right)\right) \leq c x_{uv} && \forall\, uv \in E \\
& -y_v \leq \sum_{u \in N(v)} \left(g^q_{uv} - g^q_{vu}\right) \leq y_v && \forall\, v \in V, \forall\, q \in Q \\
& x_{uv} \in \mathbb{Z}_{\geq 0}, y_v \in \{0, 1\}, f^q_{uv} \geq 0, g^q_{uv} \geq 0
\end{aligned}
$$

NDPC as MILP

$$
\begin{aligned}
\min \; & \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v \\
\text{s.t.} \; & \sum_{u \in N(v)} \left(f^q_{vu} + g^q_{vu} - f^q_{uv} - g^q_{uv} \right) = \begin{cases} -1 & \text{if } u = s^q, \\ 1 & \text{if } u = t^q, \\ 0 & \text{else} \end{cases} && \forall\, v \in V, \forall\, q \in Q \\
& \sum_{q \in Q} \left(d^q \left(f^q_{uv} + f^q_{vu} \right) + \lambda d^q \left(g^q_{uv} + g^q_{vu} \right) \right) \le c x_{uv} && \forall\, uv \in E \\
& - y_v \le \sum_{u \in N(v)} \left(g^q_{uv} - g^q_{vu} \right) \le y_v && \forall\, v \in V, \forall\, q \in Q \\
& x_{uv} \in \mathbb{Z}_{\ge 0}, y_v \in \{0,1\}, f^q_{uv} \ge 0, g^q_{uv} \ge 0
\end{aligned}
$$

$$\min \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v$$

$$\text{s.t.} \sum_{u \in N(v)} \left(f^q_{vu} + g^q_{vu} - f^q_{uv} - g^q_{uv}\right) = \begin{cases} -1 & \text{if } u = s^q, \\ 1 & \text{if } u = t^q, \\ 0 & \text{else} \end{cases} \qquad \forall\, v \in V, \forall\, q \in Q$$

$$\sum_{q \in Q} \left(d^q \left(f^q_{uv} + f^q_{vu}\right) + \lambda d^q \left(g^q_{uv} + g^q_{vu}\right)\right) \leq c x_{uv} \qquad \forall\, uv \in E$$

$$-y_v \leq \sum_{u \in N(v)} \left(g^q_{uv} - g^q_{vu}\right) \leq y_v \qquad \forall\, v \in V, \forall\, q \in Q$$

$$x_{uv} \in \mathbb{Z}_{\geq 0}, y_v \in \{0, 1\}, f^q_{uv} \geq 0, g^q_{uv} \geq 0$$

$$
\begin{aligned}
\min & \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v \\
\text{s.t.} & \sum_{u \in N(v)} \left(f^q_{vu} + g^q_{vu} - f^q_{uv} - g^q_{uv} \right) = \begin{cases} -1 & \text{if } u = s^q, \\ 1 & \text{if } u = t^q, \\ 0 & \text{else} \end{cases} & \forall\, v \in V, \forall\, q \in Q \\
& \sum_{q \in Q} \left(d^q \left(f^q_{uv} + f^q_{vu} \right) + \lambda d^q \left(g^q_{uv} + g^q_{vu} \right) \right) \leq c x_{uv} & \forall\, uv \in E \\
& - y_v \leq \sum_{u \in N(v)} \left(g^q_{uv} - g^q_{vu} \right) \leq y_v & \forall\, v \in V, \forall\, q \in Q \\
& x_{uv} \in \mathbb{Z}_{\geq 0}, y_v \in \{0,1\}, f^q_{uv} \geq 0, g^q_{uv} \geq 0
\end{aligned}
$$

How difficult is Network Design with Compression?

Compression vs. No Compression

How difficult is Network Design with Compression?

instance	abilene	germany
$\|V\|$	12	17
$\|E\|$	15	26
$\|Q\|$	130	251

CPU time to optimality

How difficult is Network Design with Compression?

instance	abilene	germany	factor
$\|V\|$	12	17	
$\|E\|$	15	26	
$\|Q\|$	130	251	
w/o compression	0.14 s	5.82 s	
with compression	19.95 s	2,219.25 s	> 142

CPU time to optimality

How difficult is Network Design with Compression?

instance	abilene	germany	factor
$\lvert V\rvert$	12	17	
$\lvert E\rvert$	15	26	
$\lvert Q\rvert$	130	251	
w/o compression	0.14 s	5.82 s	
with compression	19.95 s	2,219.25 s	> 142

CPU time to optimality

Compression vs. No Compression

How difficult is Network Design with Compression?

instance	abilene	germany	factor
$\lvert V \rvert$	12	17	
$\lvert E \rvert$	15	26	
$\lvert Q \rvert$	130	251	
w/o compression	0.14 s	5.82 s	
with compression	19.95 s	2,219.25 s	> 142

CPU time to optimality

Conclusion: Complexity increases significantly!

Cost of links: C_{uv}
Objective: $\min \sum_{uv \in E} C_{uv} x_{uv}$

Network Design w/o Compression

The NETWORK DESIGN problem is NP-hard.

Cost of links: C_{uv}
Objective: $\min \sum_{uv \in E} C_{uv} x_{uv}$

Network Design w/o Compression

The Network Design problem is NP-hard.

Cost of compression at node v: C_v
Objective: $\min \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v$

Network Design with Compression

Corollary: Network Design with Compression is NP-hard as well.

Complexity of the General Problem

Cost of links: C_{uv}
Objective: $\min \sum_{uv \in E} C_{uv} x_{uv}$

Network Design w/o Compression

The NETWORK DESIGN problem is NP-hard.

Cost of compression at node v: C_v
Objective: $\min \sum_{uv \in E} C_{uv} x_{uv} + \sum_{v \in V} C_v y_v$

Network Design with Compression

Corollary: NETWORK DESIGN WITH COMPRESSION is NP-hard as well.

but, is it "more difficult" than NETWORK DESIGN?

Easy case (w/o Compression!):

If G is a tree, Network Design can be solved in polynomial time.

Network Design on a Tree

Easy case (w/o Compression!):

If G is a tree, Network Design can be solved in polynomial time.

Observation: Routing is fixed

Easy case (w/o Compression!):

If G is a tree, Network Design can be solved in polynomial time.

Observation: Routing is fixed

$\hookrightarrow$ Simple rounding of flows yields required capacities.

Easy case (w/o Compression!):

If G is a tree, NETWORK DESIGN can be solved in polynomial time.

Observation: Routing is fixed

$\hookrightarrow$ Simple rounding of flows yields required capacities.

What impact does compression have?

Network Design on a Tree

Easy case (w/o Compression!):

If G is a tree, NETWORK DESIGN can be solved in polynomial time.

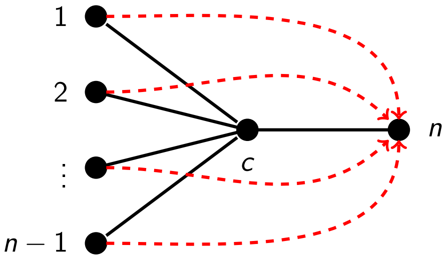
Observation: Routing is fixed

$\hookrightarrow$ Simple rounding of flows yields required capacities.

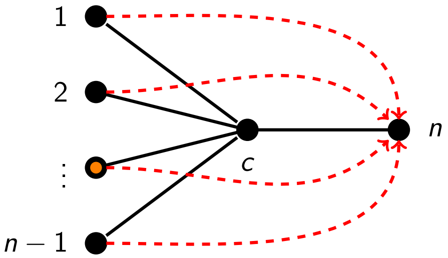
What impact does compression have?

Definition

Given a fixed routing, the **compressor placement problem** is to determine the active compressors and link capacities at minimum cost.



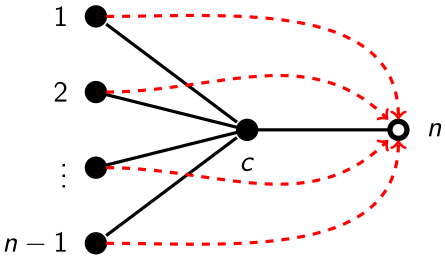
- Star G with $n+1$ vertices
- Demands d_{in}, $i = 1, \ldots, n-1$
- Capacity of c on every link (very expensive to install more)



- Star G with $n+1$ vertices
- Demands d_{in}, $i = 1, \ldots, n-1$
- Capacity of c on every link (very expensive to install more)

Observations

- If $d_{in} > c$, then $y_i = 1$ (Compression needed in i)

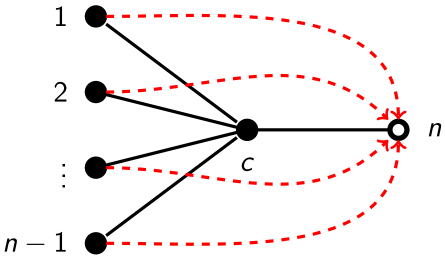


- Star G with $n+1$ vertices
- Demands d_{in}, $i = 1, \ldots, n-1$
- Capacity of c on every link (very expensive to install more)

Observations

- If $d_{in} > c$, then $y_i = 1$ (Compression needed in i) $\rightarrow d_{in} \leq c$ w.l.o.g.

Network Design on a Star

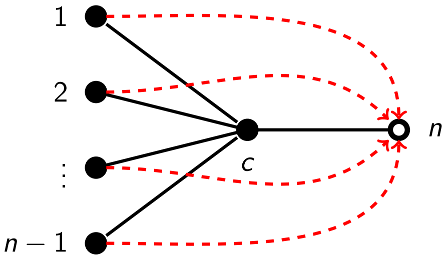


- Star G with $n+1$ vertices
- Demands d_{in}, $i = 1, \ldots, n-1$
- Capacity of c on every link (very expensive to install more)

Observations

- If $d_{in} > c$, then $y_i = 1$ (Compression needed in i) $\rightarrow d_{in} \leq c$ w.l.o.g.
- If $\sum_{i=1}^{n-1} d_{in} > c$, then at least two compressors needed

Network Design on a Star



- Star G with $n+1$ vertices
- Demands d_{in}, $i = 1, \dots, n-1$
- Capacity of c on every link (very expensive to install more)

Observations

- If $d_{in} > c$, then $y_i = 1$ (Compression needed in i) $\rightarrow d_{in} \leq c$ w.l.o.g.
- If $\sum_{i=1}^{n-1} d_{in} > c$, then at least two compressors needed
  Where to place converters? In center-node, or individual nodes $1, \dots, n-1$?

Theorem

Network Design with Compression *on stars is at least weakly NP-hard.*

Proof:

Theorem

NETWORK DESIGN WITH COMPRESSION *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

- profits c_i, $i \in N := \{1, \ldots, n-1\}$
- weights a_i, $i \in N$
- capacity B with $\max_{i \in N} a_i \leq B$ and $\sum_{i \in N} a_i > B$

Theorem

NETWORK DESIGN WITH COMPRESSION *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

- profits c_i, $i \in N := \{1, \ldots, n-1\}$
- weights a_i, $i \in N$
- capacity B with $\max_{i \in N} a_i \leq B$ and $\sum_{i \in N} a_i > B$

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges

Theorem

Network Design with Compression on stars is at least weakly NP-hard.

Proof: Reduction from Knapsack

- profits c_i, $i \in N := \{1, \dots, n-1\}$
- weights a_i, $i \in N$
- capacity B with $\max_{i \in N} a_i \leq B$ and $\sum_{i \in N} a_i > B$

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$

Theorem

Network Design with Compression on stars is at least weakly NP-hard.

Proof: Reduction from Knapsack

- profits c_i, $i \in N := \{1, \dots, n-1\}$
- weights a_i, $i \in N$
- capacity B with $\max_{i \in N} a_i \leq B$ and $\sum_{i \in N} a_i > B$

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$

Network Design on a Star

Theorem

Network Design with Compression *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$
4. Set $\lambda = \frac{1}{2}$, $c = \lambda \sum_{i \in N} a_i + (1 - \lambda) B$

Theorem

NETWORK DESIGN WITH COMPRESSION on stars is at least weakly NP-hard.

Proof: Reduction from Knapsack

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$
4. Set $\lambda = \frac{1}{2}$, $c = \lambda \sum_{i \in N} a_i + (1 - \lambda) B$
5. Baseline solution: $y_i = 1 \quad \forall i \in N \cup \{n\}$, $x_{ij} = 1 \quad \forall ij \in E$

Network Design on a Star

Theorem

Network Design with Compression *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$
4. Set $\lambda = \frac{1}{2}$, $c = \lambda \sum_{i \in N} a_i + (1 - \lambda) B$
5. Baseline solution: $y_i = 1 \quad \forall i \in N \cup \{n\}$, $x_{ij} = 1 \quad \forall ij \in E$
6. Min Cost = Max Savings to baseline solution

Theorem

NETWORK DESIGN WITH COMPRESSION *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$
4. Set $\lambda = \frac{1}{2}$, $c = \lambda \sum_{i \in N} a_i + (1 - \lambda)B$
5. Baseline solution: $y_i = 1 \quad \forall i \in N \cup \{n\}$, $x_{ij} = 1 \quad \forall ij \in E$
6. Min Cost = Max Savings to baseline solution
7. Spare capacity on link cn: $(1 - \lambda)B$

   Extra flow by removing compression at node i: $(1 - \lambda)a_i$

Theorem

NETWORK DESIGN WITH COMPRESSION *on stars is at least weakly NP-hard.*

Proof: Reduction from Knapsack

1. Let $G = (V, E)$ with $V := N \cup \{n, c\}$, E star edges
2. Set $C_i := \begin{cases} c_i & i \in N \\ 0 & i = n \\ \infty & i = c \end{cases}$ 3. Set $C_{ij} := \begin{cases} 0 & i \in N, j = c \\ M & i = c, j = n \end{cases}$ with $M > \sum C_i$
4. Set $\lambda = \frac{1}{2}$, $c = \lambda \sum_{i \in N} a_i + (1 - \lambda)B$
5. Baseline solution: $y_i = 1 \quad \forall i \in N \cup \{n\}$, $x_{ij} = 1 \quad \forall ij \in E$
6. Min Cost = Max Savings to baseline solution
7. Spare capacity on link cn: $(1 - \lambda)B$

   Extra flow by removing compression at node i: $(1 - \lambda)a_i$
8. Max Savings = Max Knapsack $\square$

Theorem

Network Design with Compression *is strongly NP-hard*

Proof:

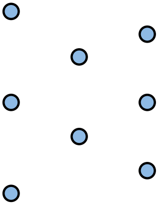
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \ \forall i = 1, \ldots, n$?



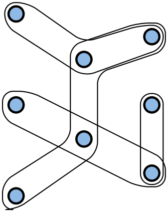
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \; \forall i = 1, \ldots, n$?



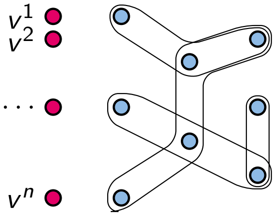
Idea suggested by Stéphane Perennes

Theorem

Network Design with Compression *is strongly NP-hard*

Proof:

Reduction from Hitting Set: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \; \forall i = 1, \ldots, n$?



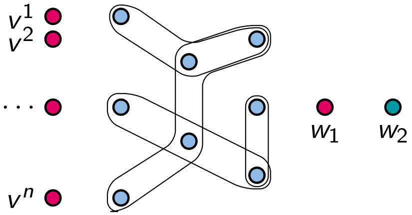
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \; \forall i = 1, \ldots, n$?



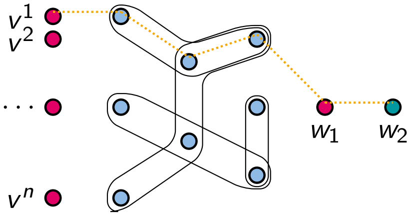
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \ \forall i = 1, \ldots, n$?



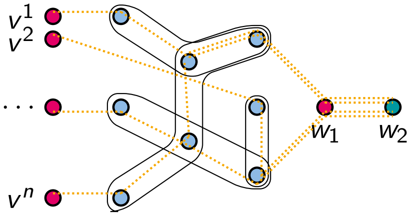
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \ \forall i = 1, \ldots, n$?



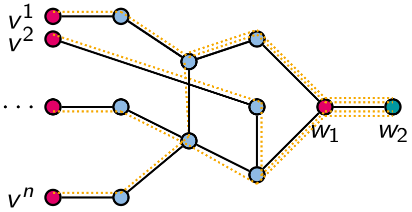
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \; \forall i = 1, \ldots, n$?



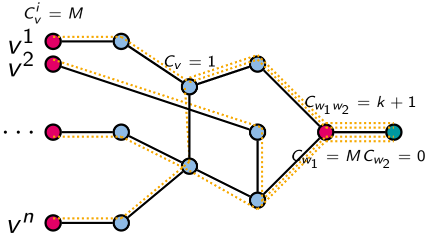
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \; \forall i = 1, \ldots, n$?



capacity $c = 1$

demand $d_{v^i w_2} = \frac{2}{n}$

$\lambda = \frac{1}{2}$

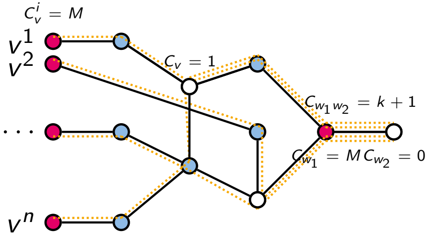
Idea suggested by Stéphane Perennes

Theorem

NETWORK DESIGN WITH COMPRESSION *is strongly NP-hard*

Proof:

Reduction from HITTING SET: "Universe" U, subsets $S_i \subseteq U$, and integer k, $\exists H \subseteq U$ with $|H| \leq k$ such that $H \cap S_i \neq \emptyset \ \forall i = 1, \ldots, n$?



capacity $c = 1$

demand $d_{v^i w_2} = \frac{2}{n}$

$\lambda = \frac{1}{2}$

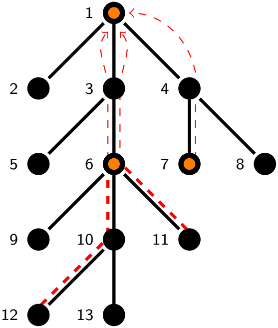
$|H| \leq k$ if and only if Cost of NDPC $\leq 2k + 1$

Idea suggested by Stéphane Perennes

Theorem

Network Design with Compression *on trees is weakly NP-hard.*

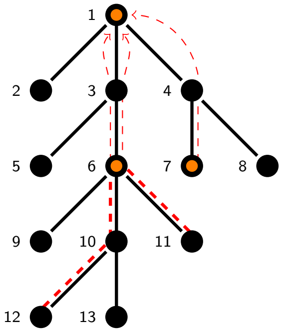
- Tree $G = (V, E)$, $|V| = n$, nodes labeled increasingly by BFS, starting at $i = 1$



Theorem

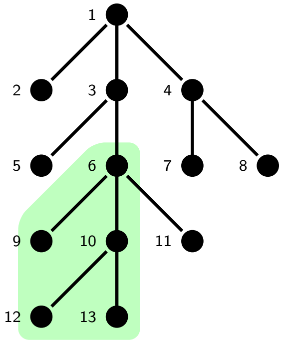
Network Design with Compression *on trees is weakly NP-hard.*

- Tree $G = (V, E)$, $|V| = n$, nodes labeled increasingly by BFS, starting at $i = 1$
- Capacity $c \in \mathbb{Z}_{\geq 0}$ per installed batch
- Commodities $Q = \{(i, 1) : i \geq 2\}$, $d_i \in \mathbb{Z}_{\geq 0}$, direct routing



Notation

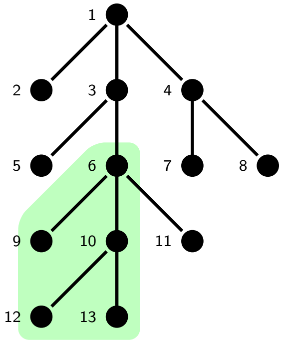
- $[i,k]$ subtree induced by i and offspring of i's first k children



subtree $[6,2]$

Notation

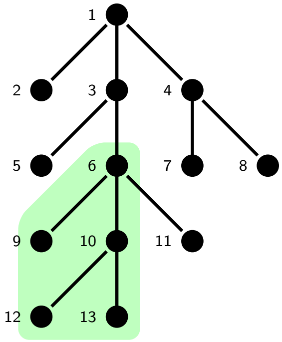
- $[i, k]$ subtree induced by i and offspring of i's first k children
- $p(i)$ predecessor of $i \neq 1$, $s(i, k)$ sibling k of i



subtree $[6, 2]$

Notation

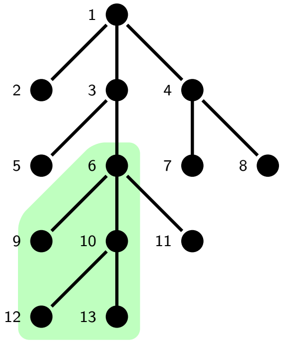
- $[i,k]$ subtree induced by i and offspring of i's first k children
- $p(i)$ predecessor of $i \neq 1$, $s(i,k)$ sibling k of i
- $a(i)$ number of children of i, $a(i,k) := a(s(i,k))$



subtree $[6,2]$

Notation

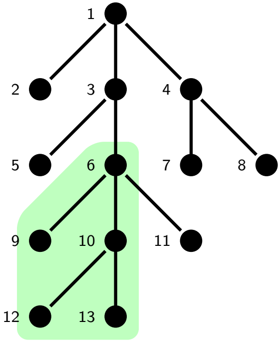
- $[i, k]$ subtree induced by i and offspring of i's first k children
- $p(i)$ predecessor of $i \neq 1$, $s(i, k)$ sibling k of i
- $a(i)$ number of children of i, $a(i, k) := a(s(i, k))$
- $d([i, k])$ demand induced by subgraph $[i, k]$



subtree $[6, 2]$

Cost functions:

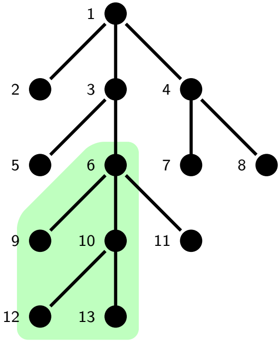
- $\mathcal{C}([i,k],f)$: min cost of $[i,k]$ with **compressing** in i and uncompressed flow of f on $(i,p(i))$ (but cost not counted yet)
- $\mathcal{D}([i,k],f)$: min cost of $[i,k]$ with **decompressing** in i and uncompressed flow of f on $(i,p(i))$
- $\mathcal{N}([i,k],f)$: min cost of $[i,k]$ with **neither** compressing nor decompressing and uncompressed flow of f on $(i,p(i))$



subtree $[6,2]$

Cost functions:

- $\mathcal{C}([i,k],f)$: min cost of $[i,k]$ with **compressing** in i and uncompressed flow of f on $(i,p(i))$ (but cost not counted yet)
- $\mathcal{D}([i,k],f)$: min cost of $[i,k]$ with **decompressing** in i and uncompressed flow of f on $(i,p(i))$
- $\mathcal{N}([i,k],f)$: min cost of $[i,k]$ with **neither** compressing nor decompressing and uncompressed flow of f on $(i,p(i))$



subtree $[6,2]$

Lemma

Given a tree instance, an optimal solution of NDPC *is given by*
$\min\{\mathcal{D}([1,a(1)],0),\mathcal{N}([1,a(1)],0)\}$.

Observation: For $i \neq 1$, only $\mathcal{C}([i,k], 0)$ and $\mathcal{D}([i,k], d[i,k])$ needed.

Observation: For $i \neq 1$, only $\mathcal{C}([i,k],0)$ and $\mathcal{D}([i,k], d[i,k])$ needed.

Lemma (Initialization)

For $i \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is

- $\mathcal{C}([i,0],0) = C_i$,
- $\mathcal{D}([i,a(i)], d([i,0])) = C_i$,
- $\mathcal{N}([i,0],f) = 0$.

Observation: For $i \neq 1$, only $\mathcal{C}([i,k],0)$ and $\mathcal{D}([i,k],d[i,k])$ needed.

Lemma (Initialization)

For $i \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is

- $\mathcal{C}([i,0],0) = C_i$,
- $\mathcal{D}([i,a(i)],d([i,0])) = C_i$,
- $\mathcal{N}([i,0],f) = 0$.

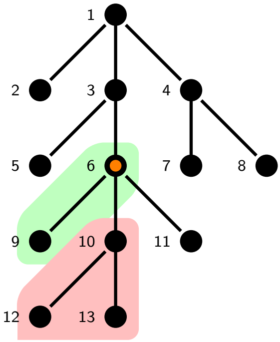
Observation: For $i \neq 1$, only $\mathcal{C}([i,k],0)$ and $\mathcal{D}([i,k],d[i,k])$ needed.

Lemma (Initialization)

For $i \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is

- $\mathcal{C}([i,0],0) = C_i$,
- $\mathcal{D}([i,a(i)],d([i,0])) = C_i$,
- $\mathcal{N}([i,0],f) = 0$.

Observation: For $i \neq 1$, only $\mathcal{C}([i,k],0)$ and $\mathcal{D}([i,k],d[i,k])$ needed.



Lemma (Initialization)

For $i \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is

- $\mathcal{C}([i,0],0) = C_i$,
- $\mathcal{D}([i,a(i)],d([i,0])) = C_i$,
- $\mathcal{N}([i,0],f) = 0$.

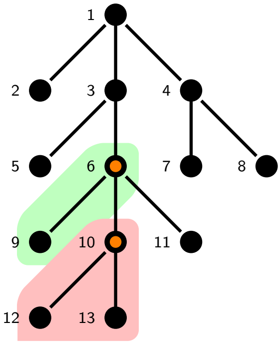
Lemma (Recursion $\mathcal{C}$ompression)

Let $x_0 := d([s(i,k),a(i,k)])$. For every node $i \neq 1$ and $k = 1,\dots,a(i)$, it is

$$\mathcal{C}([i,k],0) = \mathcal{C}([i,k-1],0) + \min \left\{ \begin{array}{l} \mathcal{C}([s(i,k),a(i,k)],0) + C_{s(i,k)i}\lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)},\dots,x_0\}} \left\{ \begin{array}{l} \mathcal{N}([s(i,k),a(i,k)],x) \\ +C_{s(i,k)i}\lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}.$$

Network Design on a Tree

Observation: For $i \neq 1$, only $\mathcal{C}([i,k],0)$ and $\mathcal{D}([i,k],d[i,k])$ needed.



Lemma (Initialization)

For $i \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is

- $\mathcal{C}([i,0],0) = C_i$,
- $\mathcal{D}([i,a(i)],d([i,0])) = C_i$,
- $\mathcal{N}([i,0],f) = 0$.

Lemma (Recursion $\mathcal{C}$ompression)

Let $x_0 := d([s(i,k),a(i,k)])$. For every node $i \neq 1$ and $k = 1,\dots,a(i)$, it is

$$\mathcal{C}([i,k],0) = \mathcal{C}([i,k-1],0) + \min \left\{ \begin{array}{c} \mathcal{C}([s(i,k),a(i,k)],0) + C_{s(i,k)i} \lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)},\dots,x_0\}} \left\{ \begin{array}{c} \mathcal{N}([s(i,k),a(i,k)],x) \\ + C_{s(i,k)i} \lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}.$$

Lemma (Recursion $\mathcal{D}$ecompression)

Let $x_0 := d([s(i,k), a(i,k)])$*. For every node* $i \neq 1$ *and* $k = 1, \ldots, a(i)$*, it is*

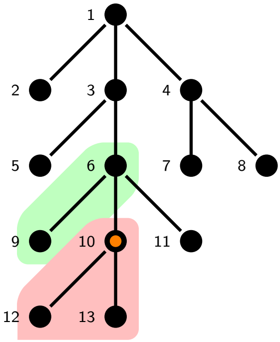
$$\mathcal{D}\left([i,k], d([i,a(i)])\right) = \mathcal{D}\left([i,a(i)], d([i,k-1])\right) + \min \left\{ \begin{array}{l} \mathcal{C}\left([s(i,k), a(i,k)], 0\right) + C_{s(i,k)i} \lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)}, \ldots, x_0\}} \left\{ \begin{array}{l} \mathcal{N}\left([s(i,k), a(i,k)], x\right) \\ + C_{s(i,k)i} \lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}$$

Network Design on a Tree

Lemma (Recursion $\mathcal{D}$ecompression)

Let $x_0 := d([s(i,k), a(i,k)])$*. For every node* $i \neq 1$ *and* $k = 1, \ldots, a(i)$*, it is*

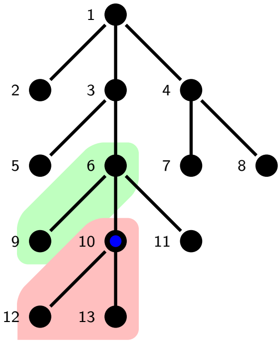
$$\mathcal{D}\left([i,k], d([i,a(i)])\right) = \mathcal{D}\left([i,a(i)], d([i,k-1])\right) + \min \left\{ \begin{array}{l} \mathcal{C}\left([s(i,k),a(i,k)],0\right) + C_{s(i,k)i}\lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)},\ldots,x_0\}} \left\{ \begin{array}{l} \mathcal{N}\left([s(i,k),a(i,k)],x\right) \\ +C_{s(i,k)i}\lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}$$



Lemma (Recursion $\mathcal{N}$either compression nor decompression)

Define $x_0 := d([s(i,k), a(i,k)])$*. For* $i \neq 1$*,* $k = 1, \ldots, a(i)$*, and for* $f = d^i \ldots, d([i,k])$*, it is*

$$\mathcal{N}\left([i,k], f\right) = \min \left\{ \begin{array}{l} \color{red}{\mathcal{N}\left([i,k-1], f\right) + \mathcal{C}\left([s(i,k),a(i,k)],0\right) + C_{s(i,k)i}\lceil \frac{x_0}{\gamma c} \rceil}, \\ \mathcal{N}\left([i,k-1], f - x_0\right) + \mathcal{D}\left([s(i,k),a(i,k)],x_0\right) + C_{s(i,k)i}\lceil \frac{x_0}{c} \rceil, \\ \min\limits_{x \in \{d(s(i,k)),\ldots,f-d(i)\}} \left\{ \begin{array}{l} \mathcal{N}\left([i,k-1], f - x\right) + \mathcal{N}\left([s(i,k),a(i,k)],x\right) \\ \qquad +C_{s(i,k)i}\left\lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \right\rceil \end{array} \right\} \end{array} \right\}$$



Lemma (Recursion $\mathcal{D}$ecompression)

Let $x_0 := d([s(i,k), a(i,k)])$. *For every node* $i \neq 1$ *and* $k = 1, \ldots, a(i)$, *it is*

$$\mathcal{D}\left([i,k], d([i,a(i)])\right) = \mathcal{D}\left([i,a(i)], d([i,k-1])\right) + \min \left\{ \begin{array}{l} \mathcal{C}\left([s(i,k), a(i,k)], 0\right) + C_{s(i,k)i} \lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)}, \ldots, x_0\}} \left\{ \begin{array}{l} \mathcal{N}\left([s(i,k), a(i,k)], x\right) \\ + C_{s(i,k)i} \lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}$$

Lemma (Recursion $\mathcal{N}$either compression nor decompression)

Define $x_0 := d([s(i,k), a(i,k)])$. *For* $i \neq 1$, $k = 1, \ldots, a(i)$, *and for* $f = d^i \ldots, d([i,k])$, *it is*

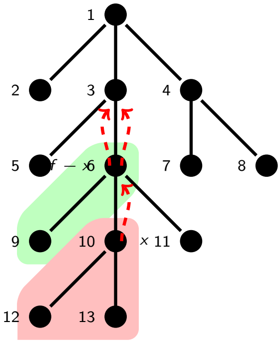
$$\mathcal{N}\left([i,k], f\right) = \min \left\{ \begin{array}{l} \mathcal{N}\left([i,k-1], f\right) + \mathcal{C}\left([s(i,k), a(i,k)], 0\right) + C_{s(i,k)i} \left\lceil \frac{x_0}{\gamma c} \right\rceil, \\ \mathcal{N}\left([i,k-1], f - x_0\right) + \mathcal{D}\left([s(i,k), a(i,k)], x_0\right) + C_{s(i,k)i} \left\lceil \frac{x_0}{c} \right\rceil, \\ \min\limits_{x \in \{d(s(i,k)), \ldots, f - d(i)\}} \left\{ \begin{array}{l} \mathcal{N}\left([i,k-1], f - x\right) + \mathcal{N}\left([s(i,k), a(i,k)], x\right) \\ + C_{s(i,k)i} \left\lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \right\rceil \end{array} \right\} \end{array} \right\}$$

Network Design on a Tree

Lemma (Recursion $\mathcal{D}$ecompression)

Let $x_0 := d([s(i,k), a(i,k)])$. For every node $i \neq 1$ and $k = 1, \dots, a(i)$, it is

$$\mathcal{D}\left([i,k], d([i,a(i)])\right) = \mathcal{D}\left([i,a(i)], d([i,k-1])\right) + \min \left\{ \begin{array}{l} \mathcal{C}\left([s(i,k), a(i,k)], 0\right) + C_{s(i,k)i} \lceil \frac{x_0}{\gamma c} \rceil, \\ \min\limits_{x \in \{d^{s(i,k)}, \dots, x_0\}} \left\{ \begin{array}{l} \mathcal{N}\left([s(i,k), a(i,k)], x\right) \\ + C_{s(i,k)i} \lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \rceil \end{array} \right\} \end{array} \right\}$$



Lemma (Recursion $\mathcal{N}$either compression nor decompression)

Define $x_0 := d([s(i,k), a(i,k)])$. For $i \neq 1$, $k = 1, \dots, a(i)$, and for $f = d^i \dots, d([i,k])$, it is

$$\mathcal{N}\left([i,k], f\right) = \min \left\{ \begin{array}{l} \mathcal{N}\left([i,k-1], f\right) + \mathcal{C}\left([s(i,k), a(i,k)], 0\right) + C_{s(i,k)i} \lceil \frac{x_0}{\gamma c} \rceil, \\ \mathcal{N}\left([i,k-1], f - x_0\right) + \mathcal{D}\left([s(i,k), a(i,k)], x_0\right) + C_{s(i,k)i} \lceil \frac{x_0}{c} \rceil, \\ \min\limits_{x \in \{d(s(i,k)), \dots, f - d(i)\}} \left\{ \begin{array}{l} \mathcal{N}\left([i,k-1], f - x\right) + \mathcal{N}\left([s(i,k), a(i,k)], x\right) \\ + C_{s(i,k)i} \left\lceil \frac{x}{c} + \frac{x_0 - x}{\gamma c} \right\rceil \end{array} \right\} \end{array} \right\}$$

Let $\triangle := \max_{q \in Q} d^q$ be the maximum demand value

Theorem

NDPC *on trees can be solved in* $O(n^3 \triangle^2)$.

Proof:

Let $\triangle := \max_{q \in Q} d^q$ be the maximum demand value

Theorem

NDPC *on trees can be solved in* $O(n^3 \triangle^2)$.

Proof:

- Number of subtrees: $2n - 1$

Let $\triangle := \max_{q \in Q} d^q$ be the maximum demand value

Theorem

NDPC *on trees can be solved in* $O(n^3 \triangle^2)$.

Proof:

- Number of subtrees: $2n - 1$
- Computing one entry of $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{N}$ takes $O(n\triangle)$.
- $\mathcal{N}$ has $n\triangle$ entries per $[i, k]$

Let $\triangle := \max_{q \in Q} d^q$ be the maximum demand value

Theorem

NDPC *on trees can be solved in* $O(n^3 \triangle^2)$.

Proof:

- Number of subtrees: $2n - 1$
- Computing one entry of $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{N}$ takes $O(n\triangle)$.
- $\mathcal{N}$ has $n\triangle$ entries per $[i, k]$
- Total runtime of $O(n^3 \triangle^2)$

Train Packing Problem

Given a set of commodities $Q = \{(s^q, t^q, d^q) : q = 1, \ldots, |Q|\}$, a network $G = (V, A)$, a set of shunting yards $R \subset V$, and a train capacity C, determine the minimum number of trains needed to transport all demands, where each train can be rearranged at shunting yards R (but at least one commodity should continue with the same train).

Train Packing Problem

Given a set of commodities $Q = \{(s^q, t^q, d^q) : q = 1, \ldots, |Q|\}$, a network $G = (V, A)$, a set of shunting yards $R \subset V$, and a train capacity C, determine the minimum number of trains needed to transport all demands, where each train can be rearranged at shunting yards R (but at least one commodity should continue with the same train).

First Results: Master thesis F. Heckhausen (in preparation)

- $|A| = 1$: bin packing – NP-complete

Train Packing Problem

Given a set of commodities $Q = \{(s^q, t^q, d^q) : q = 1, \ldots, |Q|\}$, a network $G = (V, A)$, a set of shunting yards $R \subset V$, and a train capacity C, determine the minimum number of trains needed to transport all demands, where each train can be rearranged at shunting yards R (but at least one commodity should continue with the same train).

First Results: Master thesis F. Heckhausen (in preparation)

- $|A| = 1$: bin packing – NP-complete
- $G = P_n, C = 2, d^q = 1, |R| = 1$: number of trains $= \frac{(n-1)(n-2)}{2}$

A Rail Cargo Problem

Train Packing Problem

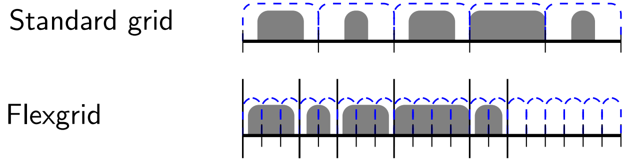
Given a set of commodities $Q = \{(s^q, t^q, d^q) : q = 1, \dots, |Q|\}$, a network $G = (V, A)$, a set of shunting yards $R \subset V$, and a train capacity C, determine the minimum number of trains needed to transport all demands, where each train can be rearranged at shunting yards R (but at least one commodity should continue with the same train).

First Results: Master thesis F. Heckhausen (in preparation)

- $|A| = 1$: bin packing – NP-complete
- $G = P_n, C = 2, d^q = 1, |R| = 1$: number of trains $= \frac{(n-1)(n-2)}{2}$
- $G = P_n, C = 2, d^q = 1, |R| = n$:

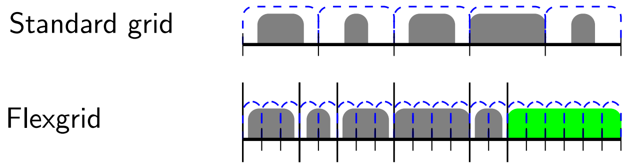
$$\text{number of trains} = \begin{cases} \frac{(n-1)(n+1)}{8} & \text{if } n = 2k+1 \\ \frac{n^2}{8} & \text{if } n = 4k \\ \frac{n^2}{8} + \frac{1}{2} & \text{if } n = 4k+2 \end{cases}$$

Idea: fixed spectrum-block size → flexible block-size



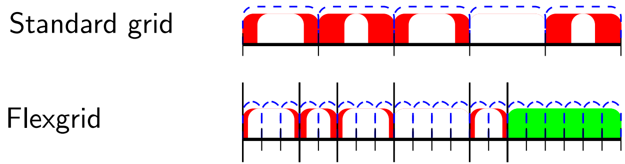
- Spectrum is divided into a small slots (e.g. 6.25GHz)

Idea: fixed spectrum-block size → flexible block-size



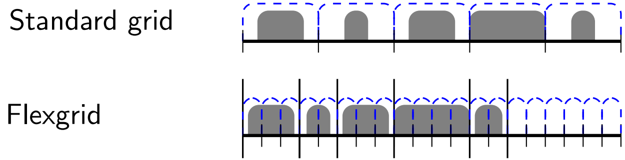
- Spectrum is divided into a small slots (e.g. 6.25GHz)
- Demands request a custom amount of these slots ('size')

Idea: fixed spectrum-block size → flexible block-size

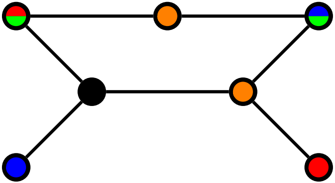


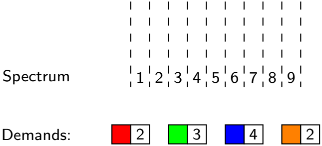
- Spectrum is divided into a small slots (e.g. 6.25GHz)
- Demands request a custom amount of these slots ('size')
  ⇒ Less spectrum wasted by custom-tailored slot sizes

Idea: fixed spectrum-block size → flexible block-size



- Spectrum is divided into a small slots (e.g. 6.25GHz)
- Demands request a custom amount of these slots ('size')
  ⇒ Less spectrum wasted by custom-tailored slot sizes
- "Freedom" is paid for: contiguity of assigned slots required



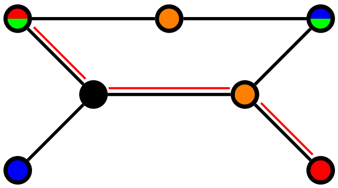


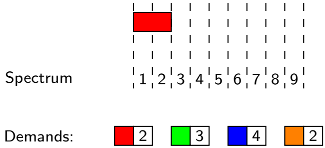
Definition (*Spectrum Allocation Problem* (SA))

Given a simple undirected graph $G = (V, E)$ and a set R of pairs $R_i = (P_i, d_i) \in \mathcal{P} \times \mathbb{N}$, $1 \leq i \leq l$, determine

1. for every R_i an interval $I_i = [a_i, b_i)$ with $a_i \leq b_i \in \mathbb{N}$ und $b_i - a_i = d_i$, such that $\max\{b_i | i = 1, \dots, l\}$ minimal, where $I_i \cap I_j = \emptyset$ if paths P_i and P_j share an edge in G.

Let $SA(G, R)$ denote the value of an optimal solution.



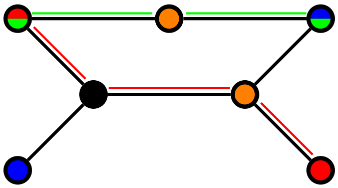


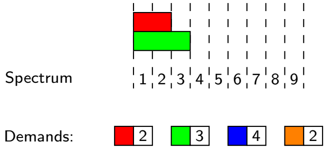
Definition (*Spectrum Allocation Problem* (SA))

Given a simple undirected graph $G = (V, E)$ and a set R of pairs $R_i = (P_i, d_i) \in \mathcal{P} \times \mathbb{N}$, $1 \leq i \leq l$, determine

1. for every R_i an interval $I_i = [a_i, b_i)$ with $a_i \leq b_i \in \mathbb{N}$ und $b_i - a_i = d_i$, such that $\max\{b_i | i = 1, \ldots, l\}$ minimal, where $I_i \cap I_j = \emptyset$ if paths P_i and P_j share an edge in G.

Let $SA(G, R)$ denote the value of an optimal solution.





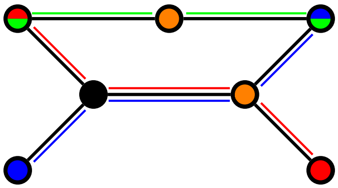
Definition (*Spectrum Allocation Problem* (SA))

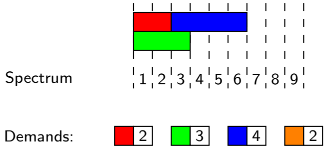
Given a simple undirected graph $G = (V, E)$ and a set R of pairs $R_i = (P_i, d_i) \in \mathcal{P} \times \mathbb{N}$, $1 \le i \le l$, determine

1. for every R_i an interval $I_i = [a_i, b_i)$ with $a_i \le b_i \in \mathbb{N}$ und $b_i - a_i = d_i$, such that $\max\{b_i | i = 1, \dots, l\}$ minimal, where $I_i \cap I_j = \emptyset$ if paths P_i and P_j share an edge in G.

Let $SA(G, R)$ denote the value of an optimal solution.

Spectrum Allocation Problem



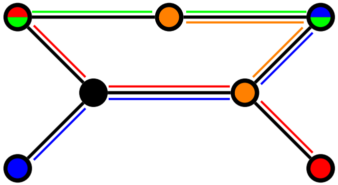


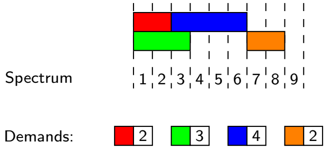
Definition (*Spectrum Allocation Problem* (SA))

Given a simple undirected graph $G = (V, E)$ and a set R of pairs $R_i = (P_i, d_i) \in \mathcal{P} \times \mathbb{N}$, $1 \leq i \leq l$, determine

1. for every R_i an interval $I_i = [a_i, b_i)$ with $a_i \leq b_i \in \mathbb{N}$ und $b_i - a_i = d_i$, such that $\max\{b_i | i = 1, \ldots, l\}$ minimal, where $I_i \cap I_j = \emptyset$ if paths P_i and P_j share an edge in G.

Let $SA(G, R)$ denote the value of an optimal solution.





Definition (*Spectrum Allocation Problem* (SA))

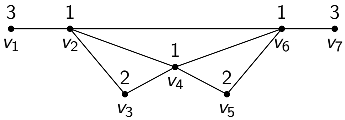
Given a simple undirected graph $G = (V, E)$ and a set R of pairs $R_i = (P_i, d_i) \in \mathcal{P} \times \mathbb{N}$, $1 \leq i \leq l$, determine

1. for every R_i an interval $I_i = [a_i, b_i)$ with $a_i \leq b_i \in \mathbb{N}$ und $b_i - a_i = d_i$, such that $\max\{b_i | i = 1, \ldots, l\}$ minimal, where $I_i \cap I_j = \emptyset$ if paths P_i and P_j share an edge in G.

Let $SA(G, R)$ denote the value of an optimal solution.

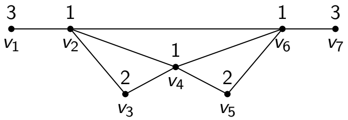
Definition (Interval-Coloring Problem (IC))

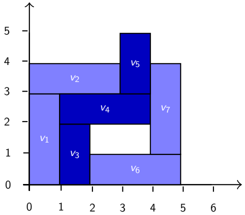
Let $G = (V, E)$ and $d : V \mapsto \mathbb{N}$. The *Interval Coloring Problem* is to assign to every vertex v an interval of length $d(v)$, such that adjacent vertices are assigned disjoint intervals. $\chi_I(G) =$ the minimum of colors required.



Definition (Interval-Coloring Problem (IC))

Let $G = (V, E)$ and $d : V \mapsto \mathbb{N}$. The *Interval Coloring Problem* is to assign to every vertex v an interval of length $d(v)$, such that adjacent vertices are assigned disjoint intervals. $\chi_I(G) =$ the minimum of colors required.





$SA(G, R, P) = \chi_I(G')$

The Spectrum Allocation Problem (G, R, P) is equivalent to the Interval-Coloring Problem on the edge-intersection graph G' of paths P_i.

Corollary

Spectrum Allocation is $\mathcal{NP}$-hard on general networks as well as on star networks

Proof for star networks: wavelength assignment ($d_i = 1$) is $\mathcal{NP}$-hard by a reduction from edge coloring.

Corollary

Spectrum Allocation is $\mathcal{NP}$-hard on general networks as well as on star networks

Proof for star networks: wavelength assignment ($d_i = 1$) is $\mathcal{NP}$-hard by a reduction from edge coloring.

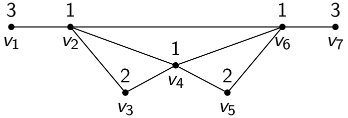
Corollary

Spectrum Allocation is already $\mathcal{NP}$-hard on path networks and $d_i \in \{1, 2\}$

Proof: Interval-Coloring on a path is equivalent to Dynamic Storage Allocation, which is known to be $\mathcal{NP}$-hard.

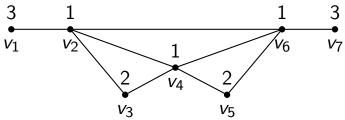
Definition (Interval-Coloring Problem (IC))

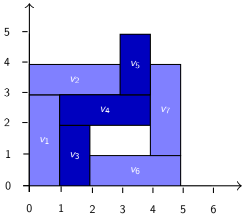
Given a graph $G = (V, E)$ and a weight function $d : V \mapsto \mathbb{N}$, the *Interval Coloring Problem* is to assign to every vertex v an interval of length $d(v)$, such that adjacent vertices are not assigned common colors. Let $\chi_I(G)$ denote the minimum of colors required.



Definition (Interval-Coloring Problem (IC))

Given a graph $G = (V, E)$ and a weight function $d : V \mapsto \mathbb{N}$, the *Interval Coloring Problem* is to assign to every vertex v an interval of length $d(v)$, such that adjacent vertices are not assigned common colors. Let $\chi_I(G)$ denote the minimum of colors required.





$SA(G, R, P) = \chi_I(G')$

The Spectrum Allocation Problem (G, R, P) is equivalent to the Interval-Coloring Problem on the edge-intersection graph G' of paths P_i.

Definition (Star)

A star $K_{1,n}$ is a graph with vertex set $V(K_{1,n}) = \{v_0, \ldots, v_n\}$ and edge set $E(K_{1,n}) = \{(v_0, v_i) | i = 1, \ldots, n)\}$.

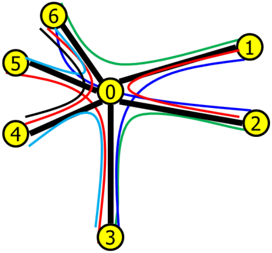
Definition (Star)

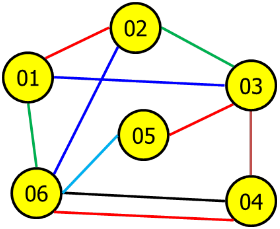
A star $K_{1,n}$ is a graph with vertex set $V(K_{1,n}) = \{v_0, \ldots, v_n\}$ and edge set $E(K_{1,n}) = \{(v_0, v_i) | i = 1, \ldots, n)\}$.

Lemma

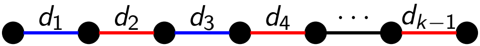
The (R)SA problem on stars is NP-hard, even if all $d_i = 1$.

Proof: Equivalent to EDGE INTERVAL-COLORING on a multigraph



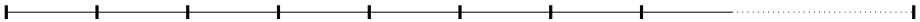


Edge Interval-Coloring on P_k

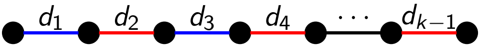


Color edges with intervals $[a_i, b_i)$ of length d_i such that

$$[a_{i-1}, b_{i-1}) \cap [a_i, b_i) = \emptyset = [a_i, b_i) \cap [a_{i+1}, b_{i+1})$$



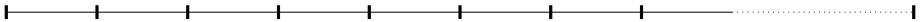
Edge Interval-Coloring on P_k



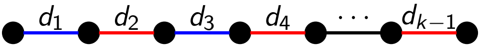
Color edges with intervals $[a_i, b_i)$ of length d_i such that

$$[a_{i-1}, b_{i-1}) \cap [a_i, b_i) = \emptyset = [a_i, b_i) \cap [a_{i+1}, b_{i+1})$$

Optimal Solution: Consider odd and even edges separately



Edge Interval-Coloring on P_k

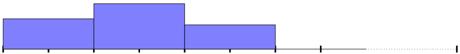


Color edges with intervals $[a_i, b_i)$ of length d_i such that

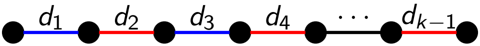
$$[a_{i-1}, b_{i-1}) \cap [a_i, b_i) = \emptyset = [a_i, b_i) \cap [a_{i+1}, b_{i+1})$$

Optimal Solution: Consider odd and even edges separately

- Assign $[0, d_{2j+1})$ for $j = 0, \ldots, \lfloor \frac{k-1}{2} \rfloor - 1$



Edge Interval-Coloring on P_k

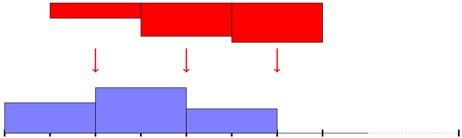


Color edges with intervals $[a_i, b_i)$ of length d_i such that

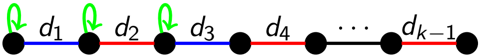
$$[a_{i-1}, b_{i-1}) \cap [a_i, b_i) = \emptyset = [a_i, b_i) \cap [a_{i+1}, b_{i+1})$$

Optimal Solution: Consider odd and even edges separately

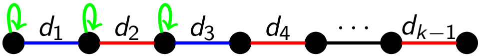
- Assign $[0, d_{2j+1})$ for $j = 0, \ldots, \lfloor \frac{k-1}{2} \rfloor - 1$
- Assign $[\chi - d_{2j}, \chi)$ for $j = 1, \ldots, \lfloor \frac{k-1}{2} \rfloor$ with $\chi := \max_{j=1,\ldots,k-2}\{d_j + d_{j+1}\}$

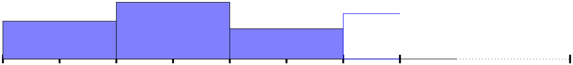


$|P_i| = 1 \Leftrightarrow$ Loop in G'

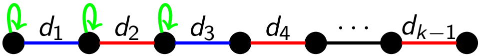


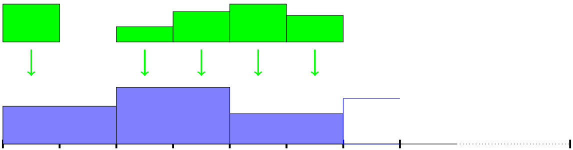
$|P_i| = 1 \Leftrightarrow$ Loop in G'



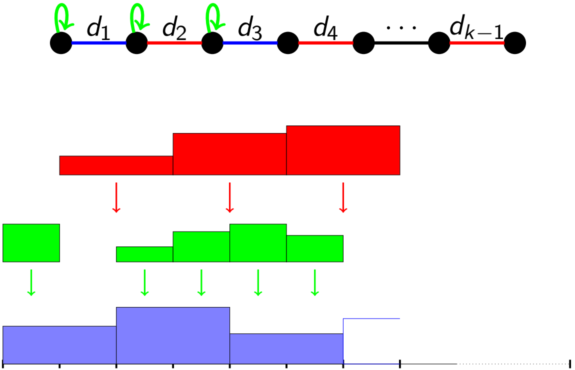


$|P_i| = 1 \Leftrightarrow$ Loop in G'

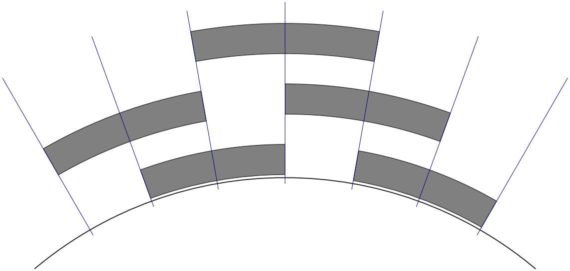




$|P_i| = 1 \Leftrightarrow$ Loop in G'



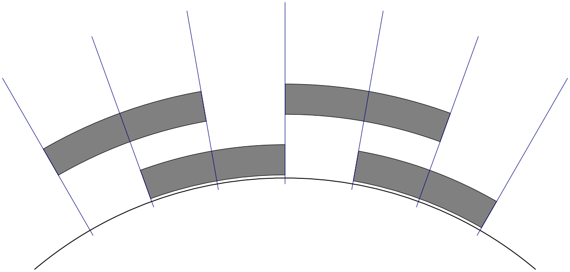
k even: analogue to paths



k even: analogue to paths

k odd:

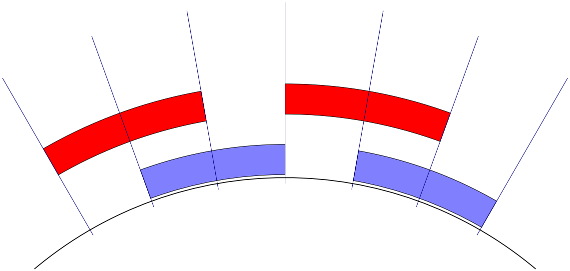
- after removal of a single edge, we obtain a path



k even: analogue to paths

k odd:

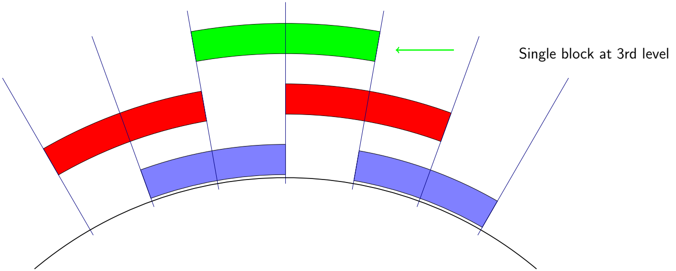
- after removal of a single edge, we obtain a path



k even: analogue to paths

k odd:

- after removal of a single edge, we obtain a path
- removed edge requires third level

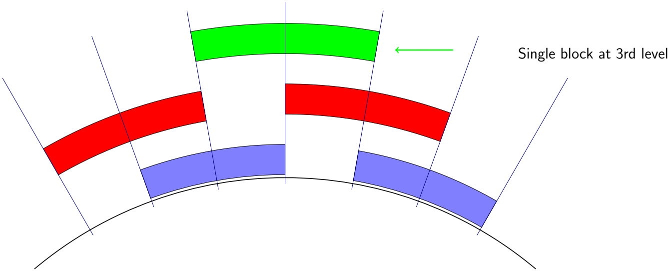


Edge Interval-Coloring on C_k

k even: analogue to paths

k odd:

- after removal of a single edge, we obtain a path
- removed edge requires third level
- search for edge (v_j, v_{j+1}) such that $d_{j-1} + d_j + d_{j+1}$ is minimized



Algorithmic Graph Theory: How hard is your combinatorial optimization problem?

Arie M.C.A. Koster

Lecture 2

Clemson, June 7, 2017

CLEMSON UNIVERSITY
Lehrstuhl II für Mathematik
RWTH AACHEN UNIVERSITY