SCIP-Jack: A Solver for Steiner Tree Problems in Graphs and their Relatives

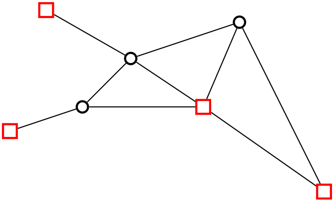
Thorsten Koch · Daniel Rehfeldt
Zuse Institute Berlin
Technical University Berlin

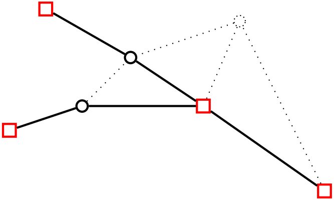
Joint Work with
Gerald Gamrath · Stephen Maher · Yuji Shinano

MIP, July, 2017

Given:

- $G = (V, E)$: undirected graph
- $T \subseteq V$: subset of vertices
- $c \in \mathbb{R}^E_{>0}$: positive edge costs





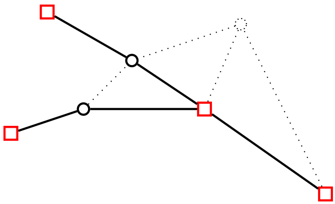
Given:

- ▷ $G = (V, E)$: undirected graph
- ▷ $T \subseteq V$: subset of vertices
- ▷ $c \in \mathbb{R}^E_{>0}$: positive edge costs

A tree $S \subseteq G$ is called Steiner tree in (G, T, c) if $T \subseteq V[S]$

Given:

- ▷ $G = (V, E)$: undirected graph
- ▷ $T \subseteq V$: subset of vertices
- ▷ $c \in \mathbb{R}^E_{>0}$: positive edge costs



A tree $S \subseteq G$ is called Steiner tree in (G, T, c) if $T \subseteq V[S]$

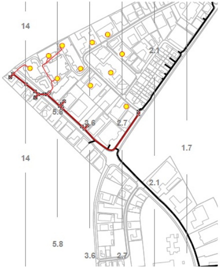
Steiner Tree Problem in Graphs (SPG)

Find a Steiner tree S in (G, T, c) with minimum edge costs $\sum_{e \in E[S]} c(e)$

SPG (decision variant) is one of Karp's 21 $\mathcal{NP}$-complete problems.

Some real-world applications of Steiner trees:

- design of fiber optic networks
- prediction of tumor evolution
- deployment of drones
- computer vision
- wire routing
- computational biology
- ...



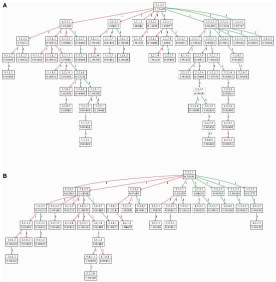
Rooted prize-collecting Steiner tree problem

E.g. *An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem* (Ljubic et al., 2006)

Applications

Some real-world applications of Steiner trees:

- ▷ design of fiber optic networks
- ▷ prediction of tumor evolution
- ▷ deployment of drones
- ▷ computer vision
- ▷ wire routing
- ▷ computational biology
- ▷ ...



Rectilinear Steiner minimum tree problem

E.g. *Phylogenetic analysis of multiprobe fluorescence in situ hybridization data from tumor cell populations* (Chowdhury et al., 2013)

Some real-world applications of Steiner trees:

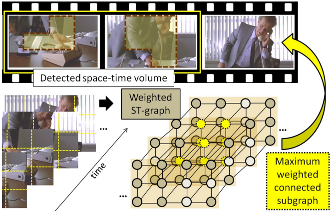
- design of fiber optic networks
- prediction of tumor evolution
- deployment of drones
- computer vision
- wire routing
- computational biology
- ...

Hop-constrained directed Steiner tree problem

E.g. *Local Search for Hop-constrained Directed Steiner Tree Problem with Application to UAV-based Multi-target Surveillance* (Burdakov, 2014)

Some real-world applications of Steiner trees:

- design of fiber optic networks
- prediction of tumor evolution
- deployment of drones
- computer vision
- wire routing
- computational biology
- ...

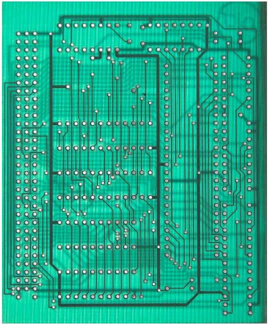


Maximum-weight connected subgraph problem

E.g. *Efficient activity detection with max-subgraph search* (Chen, Grauman, 2012)

Some real-world applications of Steiner trees:

- design of fiber optic networks
- prediction of tumor evolution
- deployment of drones
- computer vision
- write routing
- computational biology
- ...

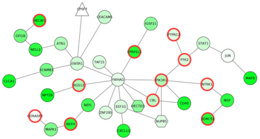


Group Steiner tree problem

E.g. *Rectilinear group Steiner trees and applications in VLSI design* (Zachariasen, Rohe, 2003)

Some real-world applications of Steiner trees:

- ▷ design of fiber optic networks
- ▷ prediction of tumor evolution
- ▷ deployment of drones
- ▷ computer vision
- ▷ wire routing
- ▷ computational biology
- ▷ ...

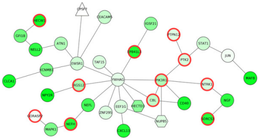


Maximum-weight connected subgraph problem

E.g. *Solving Generalized Maximum-Weight Connected Subgraph Problems for Network Enrichment Analysis* (Loboda et al., 2016)

Some real-world applications of Steiner trees:

- design of fiber optic networks
- prediction of tumor evolution
- deployment of drones
- computer vision
- wire routing
- computational biology
- ...



Maximum-weight connected subgraph problem

Real-world applications usually require variations of SPG

What we wanted: Solver for many different Steiner problem variants

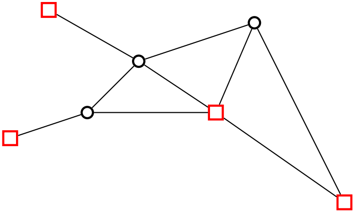
What we wanted: Solver for many different Steiner problem variants

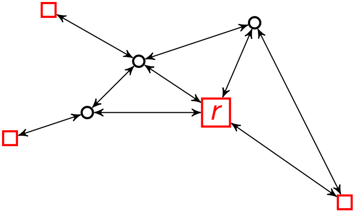
What we had: an old solver for SPG: Jack-III

What we wanted: Solver for many different Steiner problem variants

What we had: an old solver for SPG: Jack-III; based on

▷ transformation into a Steiner arborescence problem and ...

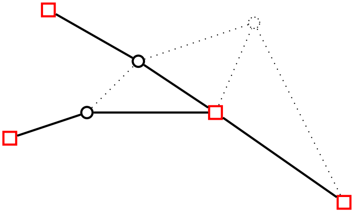


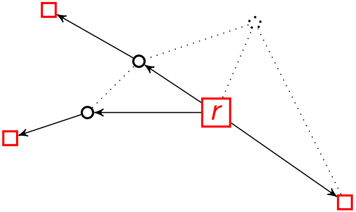


What we wanted: Solver for many different Steiner problem variants

What we had: an old solver for SPG: Jack-III; based on

- ▷ transformation into a Steiner arborescence problem and ...

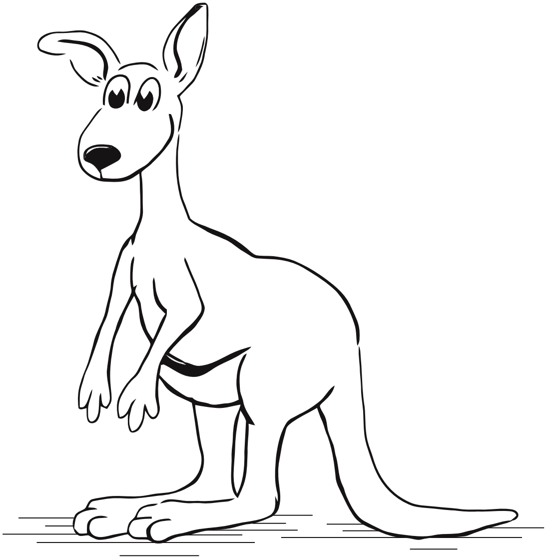




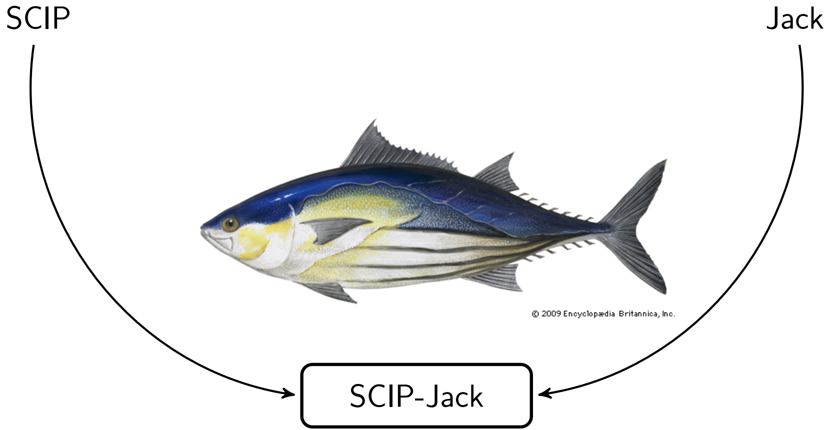
... cutting plane algorithm based on flow balance directed-cut formulation:

Formulation

$$
\begin{array}{rcll}
\min c^T y & & & \\
y(\delta_W^+) & \geqslant & 1 & \textit{for all } W \subset V, r \in W, (V \setminus W) \cap T \neq \emptyset \\
y(\delta_v^-) & \leqslant & y(\delta_v^+) & \textit{for all } v \in V \setminus T \\
y(\delta_v^-) & \geqslant & y(a) & \textit{for all } a \in \delta_v^+, v \in V \setminus T \\
y(a) & \in & \{0,1\} & \textit{for all } a \in A
\end{array}
$$



SCIP plus Jack



SCIP-Jack can solve SPG and 11 related problems:

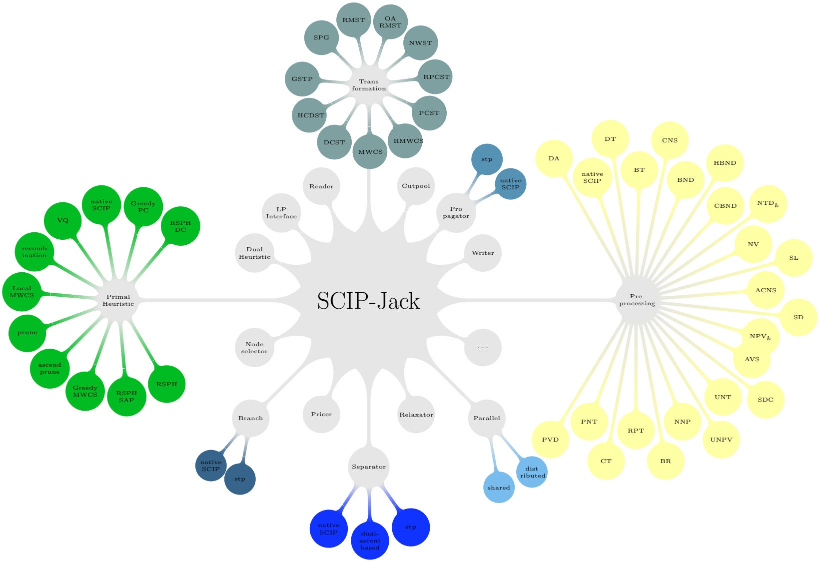
Abbreviation	Problem Name
SPG	Steiner tree problem in graphs
SAP	Steiner arborescence problem
RSMT	Rectilinear Steiner minimum tree problem
OARSMT	Obstacle-avoiding rectilinear Steiner minimum tree problem
NWSTP	Node-weighted Steiner tree problem
PCSTP	Prize-collecting Steiner tree problem
RPCSTP	Rooted prize-collecting Steiner tree problem
MWCSP	Maximum-weight connected subgraph problem
RMWCSP	Rooted maximum-weight connected subgraph problem
DCSTP	Degree-constrained Steiner tree problem
GSTP	Group Steiner tree problem
HCDSTP	Hop-constrained directed Steiner tree problem

SCIP-Jack works by combining generic and problem specific algorithms:

SCIP-Jack works by combining generic and problem specific algorithms:

- generic
  - extremly fast separator routine based on new max-flow implementation
  - all general methods provided by SCIP
    e.g., generic cutting planes and sophisticated branching
- problem specific
  - efficient transformations to Steiner arborescence problem (needed for applying generic separator)
  - preprocessing routines
  - primal and dual heuristics

Problem	Special Constraints	Virtual Vertices	Virtual Arcs	Special Preprocessing	Special Heuristics
SPG	–	–	✓	✓	✓
SAP	–	–	–	✓	✓
RSMT	–	✓	✓	–	–
OARSMT	–	✓	✓	–	–
NWSTP	–	–	✓	–	–
PCSTP	–	✓	✓	✓	✓
RPCSTP	–	✓	✓	✓	✓
MWCSP	–	✓	✓	✓	✓
RMWCSP	–	✓	✓	–	✓
DCSTP	✓	–	✓	–	✓
GSTP	–	✓	✓	–	–
HCDSTP	✓	–	–	✓	✓


SCIP-Jack
Trans formation
RMST
OA RMST
SPG
NWST
RPCST
GSTP
PCST
HCDST
DCST
MWCS
RMWCS
Reader
Cutpool
LP Interface
Pro pagator
stp
native SCIP
Dual Heuristic
Writer
Primal Heuristic
native SCIP
Greedy PC
RSPH DC
VQ
recomb ination
Local MWCS
prune
ascend prune
Greedy MWCS
RSPH SAP
RSPH
Pre processing
DA
DT
CNS
native SCIP
BT
HBND
BND
CBND
NTD$_k$
NV
SL
ACNS
SD
NPV$_k$
AVS
UNT
SDC
UNPV
NNP
BR
RPT
CT
PNT
PVD
Node selector
...
Branch
native SCIP
stp
Pricer
Relaxator
Parallel
shared
dist ributed
Separator
native SCIP
dual-ascent based
stp

Performance of SCIP-Jack for SPG

SCIP-Jack is roughly two orders of magnitude faster than Jack-III (both using CPLEX 12.6 as LP-solver).
Example: SPG test set E (20 instances, up to 62 500 edges)

Performance of SCIP-Jack for SPG

SCIP-Jack is roughly two orders of magnitude faster than Jack-III (both using CPLEX 12.6 as LP-solver).
Example: SPG test set E (20 instances, up to 62 500 edges)

- Average run time (shifted geometric mean)
  - Jack-III: 32.5 seconds
  - SCIP-Jack: 0.3 seconds
- Maximum run time (both for instance e18)
  - Jack-III: 688.3 seconds
  - SCIP-Jack: 34.1 seconds

Comparison with best free SPG solver from DIMACS competition *Mozartballs* (Fischetti et al., 2017)

		Mozartballs			SCIP-Jack		
test set	instances	solved	gap [%]	∅ time [s]	solved	gap [%]	∅ time [s]
vienna-i-adv.	85	65	0.08	314.3	**82**	**0.01**	**112.4**
E	20	20	–	9.2	20	–	**0.3**
ALUE	15	13	2.85	137.9	13	**1.90**	**21.5**
PUC	50	**12**	4.09	**1299.9**	11	**2.52**	1416.2

- ▷ 1 h time limit
- ▷ shifted geometric mean for time, arithmetic mean for gap (for unsolved instances)
- ▷ 2.3 GHz, 64 GB RAM (Mozartballs) vs. 3.2 GHz, 48 GB RAM (SCIP-Jack)
- ▷ LP-solver: CPLEX 12.6 (both)
- ▷ MIP-solver: CPLEX 12.6 (Mozartballs) vs. SCIP 4.0 (SCIP-Jack)

Performance of SCIP-Jack for SPG

But: SCIP-Jack still for most SPG instances more than five times slower than best (but not-freely available) SPG solver (Daneshmand, Polzin, 2014).

Performance of SCIP-Jack for SPG

But: SCIP-Jack still for most SPG instances more than five times slower than best (but not-freely available) SPG solver (Daneshmand, Polzin, 2014).

But but: SCIP-Jack is competitive for hard instances. By using the massively parallel extension of SCIP and 3000 cores we:

- improved primal bounds for 14 SPG benchmark instances
- solved 3 SPG benchmark instances for first time to optimality

Performance of SCIP-Jack for RPCST

SCIP-Jack is highly competitive for rooted prize-collecting Steiner tree problems (also for unrooted)

- Example: hardest test instances from DIMACS Challenge 2014 (fiber optic networks, $> 20\,000$ edges)
  - run time in first publication (Ljubic '06): > 4000 seconds (scaled)
  - best run time at DIMACS Challenge[1]: > 100 seconds
  - run time SCIP-Jack: < 0.2 seconds

[1] previous version of SCIP-Jack

Maximum-Weight Connected Subgraph Problem

Given:

- undirected graph $G = (V, E)$
- vertex weights $p \in \mathbb{R}^V$

Maximum-Weight Connected Subgraph Problem (MWCS)

Find connected subgraph $S \subseteq G$ such that $\sum_{v \in V[S]} p(v)$ is maximized

Maximum-Weight Connected Subgraph Problem

Given:

- undirected graph $G = (V, E)$
- vertex weights $p \in \mathbb{R}^V$

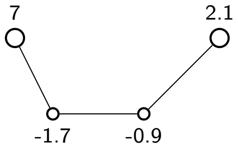
Maximum-Weight Connected Subgraph Problem (MWCS)

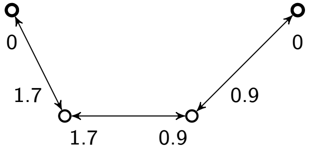
Find connected subgraph $S \subseteq G$ such that $\sum_{v \in V[S]} p(v)$ is maximized

- ...subject of many recent publications
- e.g. in computer vision and systems biology

MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

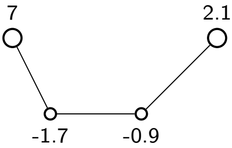
1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set
$$c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$$

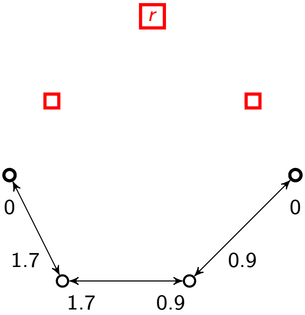




MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

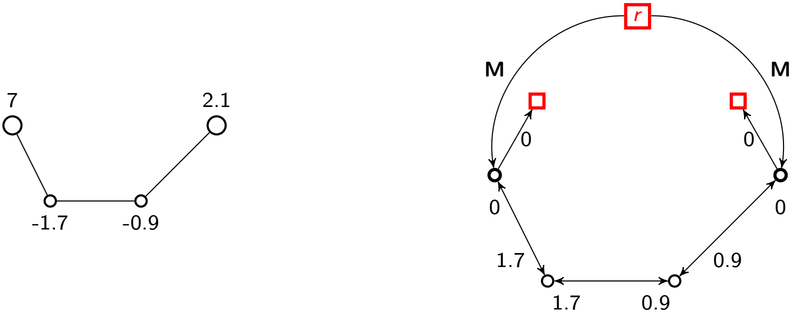
1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set
   $c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$;
   add for each t_i a terminal t_i'; add root r





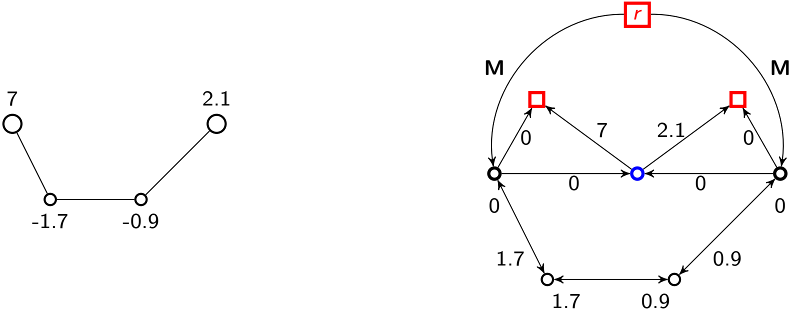
MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set $c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$; add for each t_i a terminal t'_i; add root r
3. Choose $M \geqslant \sum_{t \in T} p(t)$ $(= 9.1)$ and add arcs (r, t) of cost M for all $t \in T$



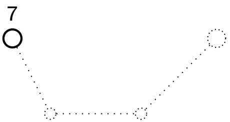
MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

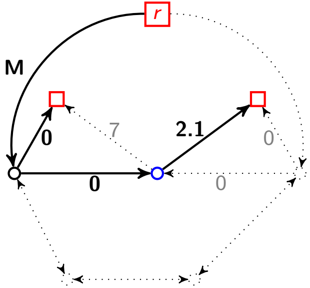
1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set
   $c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$;
   add for each t_i a terminal t'_i; add root r
3. Choose $M \geqslant \sum_{t \in T} p(t)$ $(= 9.1)$ and add arcs (r, t) of cost M for all $t \in T$
4. Add new vertex v_0, add 0-cost arcs (t_i, v_0), (t_i, t'_i) and arcs (v_0, t'_i) of cost $p(t_i)$



MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

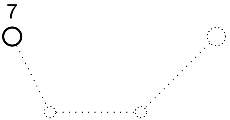
1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set
   $c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$;
   add for each t_i a terminal t'_i; add root r
3. Choose $M \geqslant \sum_{t \in T} p(t)$ $(= 9.1)$ and add arcs (r, t) of cost M for all $t \in T$
4. Add new vertex v_0, add 0-cost arcs (t_i, v_0), (t_i, t'_i) and arcs (v_0, t'_i) of cost $p(t_i)$

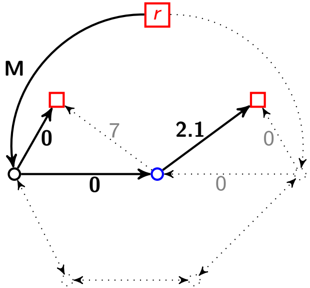




MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set $c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$; add for each t_i a terminal t_i'; add root r
3. Choose $M \geqslant \sum_{t \in T} p(t)$ $(= 9.1)$ and add arcs (r, t) of cost M for all $t \in T$
4. Add new vertex v_0, add 0-cost arcs (t_i, v_0), (t_i, t_i') and arcs (v_0, t_i') of cost $p(t_i)$

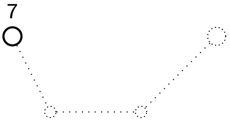


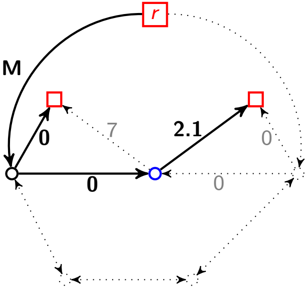


Optimal solution S' to P' corresponds to optimal solution S to P.

MWCSP $P = (V, E, p)$ is transformed to a Steiner arborescence problem $P' = (V', A', T', c')$:

1. Substitute each edge $\{v, w\}$ by two anti-parallel arcs. For each new arc $a = (v, w)$ set
$c'(a) = \begin{cases} -p(w), & \text{if } p(w) < 0 \\ 0, & \text{otherwise} \end{cases}$
2. Denote set of all $v \in V$ with $p(v) > 0$ by $T = \{t_1, ..., t_s\}$;
add for each t_i a terminal t'_i; add root r
3. Choose $M \geqslant \sum_{t \in T} p(t)$ $(= 9.1)$ and add arcs (r, t) of cost M for all $t \in T$
4. Add new vertex v_0, add 0-cost arcs (t_i, v_0), (t_i, t'_i) and arcs (v_0, t'_i) of cost $p(t_i)$





Optimal solution S' to P' corresponds to optimal solution S to P.

$$\sum_{v \in V[S]} p(v) = \sum_{v \in V: p(v) > 0} p(v) - \sum_{a \in A'[S']} c'(a) + M$$

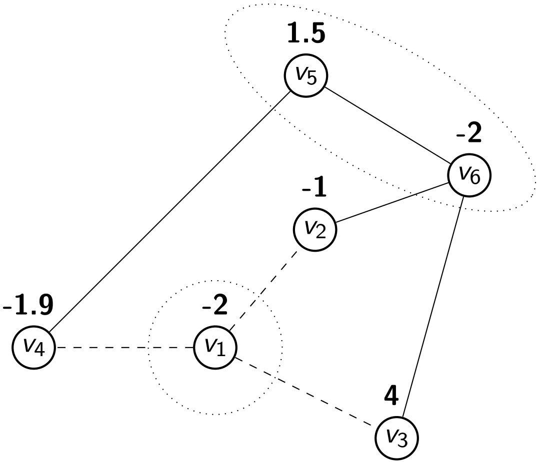
Example for MWCS reduction technique:

Lemma 1

Let $v_i \in V$ with $p(v_i) \leqslant 0$ and $W \subseteq V \setminus \{v_i\}$, $W \neq \emptyset$ such that $(W, E[W])$ is connected and $\sum_{w \in W : p(w) < 0} p(w) \geqslant p(v_i)$ holds. If

$$\Big\{v \in V \setminus W \mid \{v_i, v\} \in E\Big\} \subseteq \Big\{v \in V \setminus W \mid \{w, v\} \in E, w \in W\Big\}$$

is satisfied, then there is at least one optimal solution that does not contain v_i.



v_1 and incident edges (dashed) can be eliminated, since each neighbor of v_1 is neighbor to $W = \{v_5, v_6\}$.

- Lemma 1 is only a special case of the actual test used in SCIP-Jack, but even this special case is already NP-hard.

Reduction Techniques for MWCSP

- Lemma 1 is only a special case of the actual test used in SCIP-Jack, but even this special case is already NP-hard.
- the MWCSP reduction techniques of SCIP-Jack are empirically strong, allowing to solve many instances to optimality already during preprocessing

Reduction Techniques for MWCSP

- Lemma 1 is only a special case of the actual test used in SCIP-Jack, but even this special case is already NP-hard.
- the MWCSP reduction techniques of SCIP-Jack are empirically strong, allowing to solve many instances to optimality already during preprocessing

Comparison with best results for real-world test set *ACTMOD* from DIMACS Challenge:

Reduction Techniques for MWCSP

- Lemma 1 is only a special case of the actual test used in SCIP-Jack, but even this special case is already NP-hard.
- the MWCSP reduction techniques of SCIP-Jack are empirically strong, allowing to solve many instances to optimality already during preprocessing

Comparison with best results for real-world test set *ACTMOD* from DIMACS Challenge:

- Average run time (shifted geometric mean)
  - Best DIMACS: 4.1 seconds
  - SCIP-Jack: 0.2 seconds
- Maximum run time (for same instance)
  - Best DIMACS: 21.6 seconds
  - SCIP-Jack: 0.5 seconds

Detailed computational results for test set ACTMOD, number of vertices (V), arcs (A), and terminals (T) given for transformed SAP.

Instance	Original			Presolved					
	$\lvert V \rvert$	$\lvert A \rvert$	$\lvert T \rvert$	$\lvert V \rvert$	$\lvert A \rvert$	t [s]	Optimum	N	t [s]
drosophila001	5298	187214	72	1	0	0.2	24.3855064	1	0.2
drosophila005	5421	187952	195	24	224	0.4	178.663952	1	0.5
drosophila0075	5477	188288	251	1	0	0.3	260.523557	1	0.3
HCMV	3919	58916	56	1	0	0.1	7.55431486	1	0.1
lymphoma	2102	15914	68	1	0	0.1	70.1663087	1	0.1
metabol_expr_mice_1	3674	9590	151	1	0	0.0	544.94837	1	0.0
metabol_expr_mice_2	3600	9174	86	1	0	0.0	241.077524	1	0.0
metabol_expr_mice_3	2968	7354	115	1	0	0.0	508.260877	1	0.0

Impact of Preprocessing Methods

Results of running each MWCS reduction technique included in SCIP-Jack exhaustively on 119 instances:

Reduction Method	Removed Vertices[%]	Removed Edges[%]	∅ Time [s]
UNPV/BT	41	42	0.01
AVS	59	70	0.01
BT/NNP	88	87	0.02
NPV_k	13	10	0.01
PVD	9	11	0.00
DA	88	89	0.10
all (non-exhaustive)	99.99	99.99	0.02

Performance of SCIP-Jack

Comparision on recently published real-world computational biology test set (*SHINY*, 39 instances) with two MWCS solvers *Heinz2*/*GMWCS* (H2G) as reported in *Solving Generalized Maximum-Weight Connected Subgraph Problem for Network Enrichment Analysis* (Loboda et al., 2016):

- Average run time (shifted geometric mean)
  - H2G: > 8 seconds
  - SCIP-Jack: < 0.1 seconds
- Maximum run time
  - H2G: > 1000 seconds
  - SCIP-Jack: < 0.1 seconds

Performance of SCIP-Jack

Comparision on recently published real-world computational biology test set (*SHINY*, 39 instances) with two MWCS solvers *Heinz2*/*GMWCS* (H2G) as reported in *Solving Generalized Maximum-Weight Connected Subgraph Problem for Network Enrichment Analysis* (Loboda et al., 2016):

- Average run time (shifted geometric mean)
  - H2G: > 8 seconds
  - SCIP-Jack: < 0.1 seconds
- Maximum run time
  - H2G: > 1000 seconds
  - SCIP-Jack: < 0.1 seconds

SCIP-Jack has recently solved large-scale instance ($> 300\,000$ edges) from the 11th DIMACS Challenge for first time to optimality.

- is available as part of the SCIP Optimizations Suite
  http://scip.zib.de

- is available as part of the SCIP Optimizations Suite
  `http://scip.zib.de`
- has plugin-based structure for easy integration of further variants or solving methods

- is available as part of the SCIP Optimizations Suite
  `http://scip.zib.de`
- has plugin-based structure for easy integration of further variants or solving methods
- is highly competitive for many Steiner tree problem variants

- is available as part of the SCIP Optimizations Suite
  http://scip.zib.de
- has plugin-based structure for easy integration of further variants or solving methods
- is highly competitive for many Steiner tree problem variants
- more to come ... stay tuned

- is available as part of the SCIP Optimizations Suite
  `http://scip.zib.de`
- has plugin-based structure for easy integration of further variants or solving methods
- is highly competitive for many Steiner tree problem variants
- more to come ... stay tuned

Thank you very much!

- K., Martin, Networks, 1998
  Solving Steiner tree problems in graphs to optimality
- Gamrath, K., Maher, R., Shinano, MPC, 2017
  SCIP-Jack – A solver for STP and variants with parallelization extentions
- Shinano, et al., *Solving Open MIP Instances with ParaSCIP on Supercomputers using up to 80,000 Cores*, IEEE IPDPS, 2016
- R., K., ZR 16-36, 2016, *Transformations for the Prize-Collecting Steiner Tree Problem and the Maximum-Weight Connected Subgraph Problem to SAP*
- Maher, et al., ZR 17-12, 2017, *The SCIP Optimization Suite 4.0*,

Questions?