Stabilized Column Generation for the Rooted Delay-Constrained Steiner Tree Problem

Markus Leitner [*] Mario Ruthmair [*] Günther R. Raidl [*]

[*] Institute of Computer Graphics and Algorithms, Vienna University of Technology
Favoritenstr. 9-11, 1040 Vienna, Austria
{leitner, ruthmair, raidl}@ads.tuwien.ac.at

ABSTRACT

We consider the rooted delay-constrained Steiner tree problem which arises for example in the design of centralized multicasting networks where quality of service constraints are of concern. We present a path based integer linear programming formulation which has already been considered in the literature for the spanning tree variant. Solving its linear relaxation by column generation has so far been regarded as not competitive due to long computational times needed. In this work, we show how to significantly accelerate the column generation process using two different stabilization techniques. Computational results indicate that due to the achieved speed-up our approach outperforms so-far proposed methods.

Keywords: Network design, Stabilized column generation, Delay-constrained Steiner tree

1. INTRODUCTION

When designing a communication network with a central server broadcasting or multicasting information to all or some of the participants of the network, some applications such as video conferences require a limitation of the maximal delay from the server to each client. Beside this delay-constraint minimizing the cost of establishing the network is in most cases an important design criterion. As another example, consider a package shipment organization with a central depot guaranteeing its customers a delivery within a specified time horizon. Naturally the organization aims at minimizing the transportation costs but at the same time has to hold its promise of being in time. Such network design problems can be modeled as *rooted delay-constrained Steiner tree problem (RDCSTP)*, which is an NP-hard combinatorial optimization problem [1]. The objective is to find a minimum cost Steiner tree of a given graph with the additional constraint that the total delay along each path from a specified root node to any other required node must not exceed a given delay bound.

More formally, we are given an undirected graph $G = (V,E)$ with a set V of n nodes, a fixed root node $s \in V$, a set $T \subseteq V \setminus \{s\}$ of terminal or required nodes, a set $S = V \setminus (T \cup \{s\})$ of optional Steiner nodes, a set E of m edges, a cost function $c : E \to \mathbb{Z}^+$, a delay function $d : E \to \mathbb{Z}^+$, and a delay bound $B \in \mathbb{Z}^+$. A feasible solution to the RDCSTP is a Steiner tree $G_S = (V_S, E_S)$, $s \in V_S$, $T \subseteq V_S \subseteq V$, $E_S \subseteq E$ satisfying the constraints $\sum_{e \in P_S(t)} d_e \leq B$, $\forall t \in T$, where $P_S(t) \subseteq E$ denotes the edge set of the unique path from root s to terminal t. An optimal solution G_S^* is a feasible solution with minimum costs $c(G_S^*) = \sum_{e \in E_S} c_e$.

2. PREVIOUS & RELATED WORK

There are many recent publications dedicated to this problem and its more special variants. Several metaheuristics have been applied to the RDCSTP, such as GRASP [2, 3], path-relinking [4] and variable neighborhood search [3]. More heuristic approaches can be found for the spanning tree variant with $T = V \setminus \{s\}$, e.g. GRASP and variable neighborhood descent (VND) in [5] and ant colony optimization and variable neighborhood search in [6]. Furthermore, preprocessing methods are presented in [6] to reduce the size of the graph significantly in order to speed up the solving process. Exact methods based on integer linear programming (ILP) have been explored by Leggieri et al. [7] who describe a compact extended node-based formulation using lifted Miller-Tucker-Zemlin inequalities. Since the used Big-M inequalities usually yield rather low linear programming (LP) relaxation bounds this formulation is improved by separating directed connection cuts. Several ILP approaches for the spanning tree variant have been examined by Gouveia et al. in [8] based on a path formulation solved by two different methods. Standard column generation (CG) turns out to be computationally inefficient while a Lagrangian relaxation approach together with a fast primal heuristic exhibits better performance. A third approach reformulates the constrained shortest path problem on a layered graph and solves it using a multi commodity flow (MCF) formulation. Since the size of the layered graph and therefore the efficiency of the according model heavily depends on the number of achievable discrete delay values this approach can in practice only be used for instances with a quite restricted set of achievable delay values. Additionally a MCF model usually suffers from the huge amount of flow variables used altogether leading to a slow and memory-intensive solving process. Nevertheless solving these layered graph models turned out to be very effective on certain classes of instances.

3. PATH FORMULATION

In this section we present a path based ILP formulation for the RDCSTP which is a straightforward modification of the model discussed by Gouveia et al. [8] for the spanning tree variant of the RDCSTP. In our directed formulation we use arc set A containing an arc (s, j) for each edge $\{sj\} \in E$ incident to the root node and two oppositely directed arcs (i, j), (j,i) for all other edges $\{ij\} \in E$, $i, j \neq s$. We further assume the edge cost and delay functions to be defined on the set of arcs too, i.e. $c_{ij} = c_e$ and $d_{ij} = d_e$, $\forall (i, j) \in A, e = \{ij\} \in E$. The *integer master problem* (IMP) defined by (1)–(6) is based on variables $x_{ij} \in \{0,1\}$, $\forall (i, j) \in A$, which indicate arcs included in the directed solution. We further use path variables $\lambda_p \in \{0,1\}$, $\forall p \in P = \bigcup_{t \in T} P_t$, where $P_t \subseteq 2^A$ is the set of feasible paths from the root node s to terminal t. Each path is represented by its arc set. A path $p \in P_t$ to terminal $t \in T$ is feasible if and only if it satisfies the delay bound, i.e.

$\sum_{(i,j)\in p} d_{ij} \leq B$. Variable λ_p is set to one if path $p \in P$ is realized.

$$\text{(IMP)} \quad \min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{p\in P_t} \lambda_p \geq 1 \qquad \forall t \in T \tag{2}$$

$$x_{ij} - \sum_{p\in P_t \mid (i,j)\in p} \lambda_p \geq 0 \qquad \forall t \in T,\ \forall (i,j) \in A \tag{3}$$

$$\sum_{(i,j)\in A} x_{ij} \leq 1 \qquad \forall j \in V \tag{4}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{5}$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in P \tag{6}$$

Since the number of feasible paths for each terminal $t \in T$ and thus the total number of variables in the model is in general exponentially large, we apply CG – see e.g. [9, 10] – to solve the LP relaxation. We start with a small subset $\tilde{P}_t \subseteq P_t$, $\forall t \in T$, of path variables λ_p in the *restricted master problem* (RMP) where the integrality conditions on arcs (5) and paths (6) are replaced by (7) and (8), respectively. Further variables are added on demand according to the solution of the pricing subproblem.

$$x_{ij} \geq 0 \qquad \forall (i,j) \in A \tag{7}$$

$$\lambda_p \geq 0 \qquad \forall p \in P \tag{8}$$

Let $\mu_t \geq 0$, $\forall t \in T$, denote the dual variables associated to the convexity constraints (2) and $\pi^t_{ij} \geq 0$, $\forall t \in T$, $\forall (i,j) \in A$, denote the dual variables associated to the coupling constraints (3). Then the pricing subproblem is defined as

$$(t^*, p^*) = \text{argmin}_{t\in T, p\in P_t} -\mu_t + \sum_{(i,j)\in p} \pi^t_{ij}. \tag{9}$$

Hence we need to solve a resource constrained shortest path problem on a graph (V,A) with nonnegative arc costs π^t_{ij}, $\forall (i,j) \in A$, for each terminal $t \in T$. We solve each such problem in pseudo-polynomial time $O(B \cdot |A|)$ using the dynamic programming based algorithm from [8]. As long as path variables λ_p, $p \in P_t, t \in T$ with negative reduced costs $\bar{c} = -\mu_t + \sum_{(i,j)\in p} \pi^t_{ij}$ exist, we need to add at least one of them and resolve the RMP. This process is repeated until no further variable with negative reduced costs exists.

In each iteration we add for each terminal $t \in T$ multiple path variables using the approach from [8]: We consider all nodes $v \in V$ that are adjacent to terminal t and all delay bounds $b = 0, \ldots, B - d_{vt}$ for which a path from s to v in conjunction with arc (v,t) is a feasible path to t. In case a shortest path p to v of total delay b, $b = 0, \ldots, B - d_{vt}$, exists and $p' = p \cup \{(v,t)\}$ yields negative reduced costs, the corresponding variable is added to the RMP.

4. COLUMN GENERATION STABILIZATION

It is well known that basic CG approaches typically suffer from computational instabilities such as degeneracy or the tailing-off effect [11] which often increase the needed computational effort for solving them dramatically. Stabilization techniques to reduce the effects of these instabilities are usually classified into problem specific approaches such as the usage of dual-optimal inequalities [12, 13] and problem independent approaches, see e.g. [14, 15]. The latter are often based on the concept of stability centers and deviations from a current stability center are penalized, e.g. by using piecewise linear penalty functions. Recently, we showed how to significantly accelerate the CG process for a survivable network design problem without modifying the RMP by choosing alternative dual optimal solutions when solving the pricing subproblem [16, 17, 18]. In the following we will adapt this technique for the RDCSTP before we discuss an alternative stabilization approach based on piecewise linear penalty functions.

4.1. Alternative Dual Optimal Solutions

Let $\gamma_j \leq 0$, $\forall j \in V$, be the dual variables associated to constraints (4) and $\tilde{P} = \bigcup_{t\in T} \tilde{P}_t$ denote the set of paths for which corresponding variables have already been included in the RMP. Then the dual of the RMP is given by (10)–(15).

$$\max \sum_{t\in T} \mu_t + \sum_{j\in V} \gamma_j \tag{10}$$

$$\text{s.t.} \quad \sum_{t\in T} \pi^t_{ij} + \gamma_j \leq c_{ij} \qquad \forall (i,j) \in A \tag{11}$$

$$\mu_t - \sum_{(i,j)\in p} \pi^t_{ij} \leq 0 \qquad \forall t \in T,\ \forall p \in \tilde{P}_t \tag{12}$$

$$\mu_t \geq 0 \qquad \forall t \in T \tag{13}$$

$$\pi^t_{ij} \geq 0 \qquad \forall t \in T,\ \forall (i,j) \in A \tag{14}$$

$$\gamma_j \leq 0 \qquad \forall j \in V \tag{15}$$

Let (μ^*, π^*, γ^*) denote the current dual solution computed by the used LP solver when solving the RMP. It is easy to see that for arcs A' not part of any so far included path – i.e. $A' = \{(i,j) \in A \mid \nexists p \in \tilde{P} : (i,j) \in p\}$ – any values for the dual variables π_{ij} are optimal as long as $\sum_{t\in T} {\pi^t_{ij}}^* + {\gamma_j}^* \leq c_{ij}$, $\forall (i,j) \in A'$, since they do not occur in inequalities (12). Dual variable values ${\pi^t_{ij}}^*$, $t \in T$, may also be increased for arcs $(i,j) \in A \setminus A'$ if inequalities (11) are not binding. We conclude that any values $\pi^t_{ij} \geq {\pi^t_{ij}}^*$, $\forall (i,j) \in A$, $\forall t \in T$, are dual optimal if $\sum_{t\in T} \pi^t_{ij} \leq \sum_{t\in T} {\pi^t_{ij}}^* + \delta_{ij}$, $\forall (i,j) \in A$ holds, where $\delta_{ij} = c_{ij} + |\gamma_j| - \sum_{t\in T} {\pi^t_{ij}}^*$, $\forall (i,j) \in A$. Note that state-of-the-art LP solvers such as IBM CPLEX, which we use in our implementation, usually choose minimal optimal dual variable values, i.e. ${\pi^t_{ij}}^* = 0$, $\forall t \in T$, $\forall (i,j) \in A'$.

On the contrary to most other stabilization approaches we do not modify the RMP. Instead we aim to choose alternative dual optimal solutions which facilitate the generation of those path variables relevant for solving the LP relaxation of the IMP by increasing the dual variable values used as arc costs in the pricing subproblem. Obviously, we can simply split the potential increase δ_{ij} equally to all relevant dual variables, i.e. use alternative dual variables $\bar{\pi}^t_{ij} = {\pi^t_{ij}}^* + \frac{\delta_{ij}}{|T|}$, $\forall t \in T$, $\forall (i,j) \in A$. In our previous work for a survivable network design problem [16, 17, 18], however, it turned out to be beneficial to initially use different dual optimal solutions, one for each terminal t, which finally converge towards $\bar{\pi}^t_{ij}$, $\forall t \in T$, $\forall (i,j) \in A$. Hence, we consider two additional variants whose correspondingly used dual variables will be denoted as $\tilde{\pi}^t_{ij}$ and $\hat{\pi}^t_{ij}$, $\forall t \in T$, $\forall (i,j) \in A$, respectively. Equation (16) defines dual variable values $\tilde{\pi}^{t'}_{ij}$ used in the pricing subproblem when considering terminal $t' \in T$. Parameter $q \in \mathbb{N}$, $1 \leq q \leq |T|$, is initially set to one and gradually incremented by $\max\{1, \frac{|T|}{10}\}$ in case no negative reduced cost path has been found. After at most ten such major steps $\tilde{\pi}^{t'}_{ij} = \bar{\pi}^{t'}_{ij}$, for each terminal t'. Thus, we can terminate the CG process if $q = |T|$ and no path variables have been added.

$$\tilde{\pi}^{t'}_{ij} = \begin{cases} {\pi^t_{ij}}^* + \frac{\delta_{ij}}{q} & \text{if } t = t' \\ {\pi^t_{ij}}^* & \text{otherwise} \end{cases}, \forall (i,j) \in A. \tag{16}$$

Dual variable values $\hat{\pi}^t_{ij}$ correspond to $\tilde{\pi}^t_{ij}$ except for the fact that q is directly set to $|T|$ once no new negative reduced cost path can be found when using $q = 1$.

4.2. Piecewise Linear Stabilization

As mentioned above other successful stabilization techniques are often based on penalizing deviations from a current stability cen-

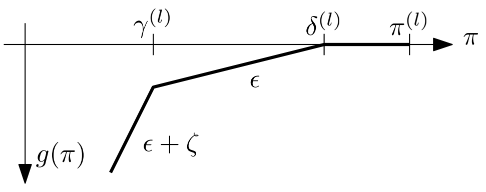

Figure 1: *Piecewise linear dual penalty function $g(\pi)$.*

ter by adding piecewise linear penalty functions to the dual problem. Among these, especially five-piecewise linear function have shown to frequently yield good results if all parameters are chosen carefully; compare [19]. In our case, however, preliminary tests with various settings and concrete parameter values showed that due to a large number of relatively time consuming updates of the stability center this concept does not seem to pay off. Since high dual variable values facilitate the generation of good path variables it is reasonable to penalize only small dual variable values. Hence we use a modified version of the approach from [14] where in each major iteration $l \in \mathbb{N}$, $l \geq 1$, only dual variable values smaller than the current stability center $\pi^{(l)} \in \mathbb{R}_+^{|T|\cdot|A|}$ are penalized according to vectors $\delta^{(l)}, \gamma^{(l)} \in \mathbb{R}_+^{|T|\cdot|A|}$, see Figure 1. Let ${\pi_{ij}^t}^*$, $\forall t \in T$, $\forall (i,j) \in A$, denote the dual variable values after the CG approach on the penalized model at major iteration l terminates. If there exists at least one dual variable value in the penalized region – i.e. if $\exists t \in T \wedge (i,j) \in A : {\pi_{ij}^t}^* < {\delta_{ij}^t}^{(l)}$ – we need to update the stability center according to the current dual solution – i.e. $\pi^{(l+1)} = \pi^*$ – and correspondingly set $\delta^{(l+1)}$ and $\gamma^{(l+1)}$ and continue the CG process. As has been shown previously [14] this process, which needs to be repeated until each dual variable value lies within an unpenalized region, terminates yielding the LP relaxation value of the IMP after finitely many steps.

According to preliminary tests, the following settings have been chosen for our computational experiments. We set $\varepsilon = 0.3$ and $\zeta = 1$, the size of the inner trust region $T^{(l)} = \pi^{(l)} - \delta^{(l)} = 1$ while $\pi^{(l)} - \gamma^{(l)} = 5 \cdot T^{(l)}$ for all dimensions, i.e. $\forall t \in T$, $\forall (i,j) \in A$. Let A' denote the set of arcs used by the paths included in the initial model. We set ${\pi_{ij}^t}^{(1)} = {\pi_{ij}^t}^* + \delta_{ij}$, $\forall t \in T$, $\forall (i,j) \in A'$, ${\pi_{ij}^t}^{(1)} = {\pi_{ij}^t}^* + \frac{\delta_{ij}}{|T|}$, $\forall t \in T, \forall (i,j) \notin A'$.

5. COMPUTATIONAL RESULTS

All computational experiments have been performed on a single core of a multi-core system consisting of Intel Xeon E5540 processors with 2.53 GHz and 3 GB RAM per core. We used IBM CPLEX 12.2 as LP solver and applied an absolute time limit of 10000 CPU-seconds to all experiments. All preprocessing methods mentioned in [6] are used to reduce the input graphs prior to solving. To build an initial set of paths a simple construction heuristic is applied on Steiner tree instances: the delay constrained shortest paths to all terminal nodes are iteratively added to the tree dissolving possible cycles. On instances where $T = V \setminus \{s\}$ we apply the Kruskal-based heuristic followed by a VND both introduced in [5]. Tables (1) and (2) report average CPU-times in seconds and needed iterations for different instance sets. In both tables π^* denotes the unstabilized CG approach, and $\bar{\pi}$, $\hat{\pi}$, and $\tilde{\pi}$ refer to the three strategies discussed in Section 4.1 for using alternative dual optimal solutions in the pricing subproblem. The piecewise linear stabilization approach from Section 4.2 is denoted by PL, Lag$_G$ and CG$_G$ denote the Lagrangian and CG approach from [8], respectively. The results of the latter two have, however, been computed on a different hardware using an older CPLEX version for the CG approach and are thus not directly comparable.

		CPU time [s]							Iterations					
Set	B	Lag$_G$	CG$_G$	π^*	$\bar{\pi}$	$\hat{\pi}$	$\tilde{\pi}$	PL	CG$_G$	π^*	$\bar{\pi}$	$\hat{\pi}$	$\tilde{\pi}$	PL
r,100	100	493	4752	314	13	15	**10**	72	1041	189	**25**	39	92	115
	150	639	8215	111	10	**8**	**8**	48	12561	357	**26**	42	98	144
	200	288	10001	123	**4**	**4**	8	46	18736	904	**28**	41	102	238
	250	526	10001	261	5	**4**	9	71	24881	1676	**32**	44	115	325
c,100	100	809	10026	38	10	**9**	12	78	480	176	**31**	44	96	171
	150	544	10034	135	26	**15**	18	142	329	346	**41**	56	118	187
	200	711	10061	1151	50	37	**21**	367	314	697	**58**	69	123	311
	250	1066	10076	3779	43	27	**25**	500	327	2702	**68**	78	141	444
e,100	100	976	10033	481	90	75	**25**	598	239	208	**40**	64	115	307
	150	1817	10106	3980	705	356	**66**	2927	193	364	**52**	84	138	403
	200	2972	10096	9297	5148	2670	**177**	8607	209	397	**92**	123	172	459
	250	4008	10104	10000	7013	3489	**142**	9090	195	357	**98**	160	203	339
r,1000	1000	971	8064	25	7	**6**	11	25	891	119	**22**	39	96	84
	1500	1744	8538	112	12	**10**	16	60	4240	253	**27**	43	112	118
	2000	869	10002	220	**11**	**11**	20	70	15600	716	**28**	42	114	125
	2500	790	10007	535	14	**12**	20	89	18233	1527	**34**	48	124	156
c,1000	1000	668	8186	60	26	24	**18**	82	869	91	**26**	38	84	109
	1500	942	10024	112	30	**25**	33	111	418	163	**37**	46	104	122
	2000	2389	10037	788	68	57	**34**	235	451	401	**36**	58	109	188
	2500	1256	10037	1272	70	**44**	48	425	437	953	**53**	62	122	261
e,1000	1000	2846	10065	137	52	34	**25**	474	615	129	**34**	56	107	165
	1500	3041	10031	4540	711	378	**71**	2787	469	266	**53**	70	130	296
	2000	5882	10083	8423	1814	897	**134**	6418	396	254	**71**	95	172	443
	2500	5726	10070	10000	4583	2222	**183**	9468	385	176	**88**	136	181	439

Table 1: Results for instance sets from [8] consisting of five complete graphs with 41 nodes, $T = V \setminus \{s\}$, different graph structures (r, c, e), delay ranges (100, 1000), and bounds B.

		CPU time [s]					Iterations				
$\frac{\lvert T\rvert}{\lvert V\setminus\{s\}\rvert}$	B	π^*	$\bar{\pi}$	$\hat{\pi}$	$\tilde{\pi}$	PL	π^*	$\bar{\pi}$	$\hat{\pi}$	$\tilde{\pi}$	PL
0.3	30	19	**6**	**6**	10	36	143	**30**	36	92	84
	50	139	**15**	16	23	55	413	**41**	50	124	102
	100	2849	97	89	**55**	509	1345	**44**	55	149	194
0.7	30	77	**29**	**29**	34	171	142	**32**	46	93	198
	50	727	112	107	**80**	1091	561	**51**	62	130	475
	100	7942	819	923	**253**	7557	1361	**79**	92	182	958
1.0	30	213	77	**62**	67	630	184	**34**	54	98	797
	50	1807	302	328	**172**	5769	614	**56**	81	142	2039
	100	9615	2615	2196	**837**	10000	851	**86**	123	214	694

Table 2: Results for 30 randomly generated complete graphs with $|V| = 100$, different sets of terminal nodes, delays and costs uniformly distributed in [1,99] and delay bounds B.

We conclude that all stabilization methods based on alternative dual-optimal solutions lead to an enormous reduction of the necessary CPU-time. While $\hat{\pi}$ performs best for easier instances, $\tilde{\pi}$ clearly outperforms all other approaches on harder instances, i.e. on those which generally need more time. Stabilization based on piecewise linear penalty functions outperforms unstabilized CG in the majority of cases, but is clearly not competitive to our three approaches based on alternative dual-optimal solutions. We further observe that our unstabilized CG variant needs significantly less iterations than the conceptually identical one discussed by Gouveia et al. [8]. We believe that next to a different CPLEX version, these differences are mainly based on choosing a better set of initial path variables, more sophisticated graph preprocessing, and the fact that we use the dual simplex algorithm which turned out to perform better than the primal one in our case. Comparing the relative computational times of the Lagrangian approach from [8] to their CG approach with the speed-up achieved by our stabilization methods, we conclude that the proposed stabilized CG method also outperforms this method. All approaches based on dual-optimal solutions terminated before the time limit was met in all but one of the experiments reported in Table 1, while both unstabilized CG variants and the piecewise linear stabilization approach failed to do so for a number of experiments.

6. CONCLUSIONS & FUTURE WORK

In this paper we showed how to significantly accelerate a column generation approach based on a path formulation for the RDC-STP using alternative dual-optimal solutions in the pricing subproblem. We conclude that this method does further outperform a stabilization method based on piecewise linear penalty functions as well as a previously presented approach based on Lagrangian relaxation [8]. We are currently extending the presented stabilized column generation towards a branch-and-price approach in order to compute proven optimal solutions to medium sized instances of the RDCSTP. In future, we also plan to consider additional pricing strategies – e.g. by restricting the total number of path variables to be included in each pricing iteration – and want to compare our approach to further stabilization techniques such as e.g. interior point stabilization [15]. Finally, we also want to study the impact of choosing better initial columns computed by metaheuristics which may lead to further significant speed-up as well as implement the Lagrangian relaxation approach from [8] for a fair comparison.

7. REFERENCES

[1] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicasting for multimedia applications," in *INFOCOM '92. Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE*, 1992, pp. 2078–2085.

[2] N. Skorin-Kapov and M. Kos, "A GRASP heuristic for the delay-constrained multicast routing problem," *Telecommunication Systems*, vol. 32, no. 1, pp. 55–69, 2006.

[3] Y. Xu and R. Qu, "A GRASP approach for the delay-constrained multicast routing problem," in *Proceedings of the 4th Multidisplinary International Scheduling Conference (MISTA4)*, Dublin, Ireland, 2009, pp. 93–104.

[4] N. Ghaboosi and A. T. Haghighat, "A path relinking approach for delay-constrained least-cost multicast routing problem," in *19th IEEE International Conference on Tools with Artificial Intelligence*, 2007, pp. 383–390.

[5] M. Ruthmair and G. R. Raidl, "A kruskal-based heuristic for the rooted delay-constrained minimum spanning tree problem," in *EUROCAST 2009*, ser. LNCS, R. Moreno-Díaz *et al.*, Eds., vol. 5717. Springer, 2009, pp. 713–720.

[6] ——, "Variable neighborhood search and ant colony optimization for the rooted delay-constrained minimum spanning tree problem," in *PPSN XI, Part II*, ser. LNCS, R. Schaefer *et al.*, Eds., vol. 6239. Springer, 2010, pp. 391–400.

[7] V. Leggieri, M. Haouari, and C. Triki, "An exact algorithm for the Steiner tree problem with delays," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 223–230, 2010.

[8] L. Gouveia, A. Paias, and D. Sharma, "Modeling and solving the rooted distance-constrained minimum spanning tree problem," *Computers & Operations Research*, vol. 35, no. 2, pp. 600–613, 2008.

[9] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, pp. 316–329, 1998.

[10] G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds., *Column Generation*. Springer, 2005.

[11] F. Vanderbeck, "Implementing mixed integer column generation," in *Column Generation*, G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds. Springer, 2005, pp. 331–358.

[12] H. B. Amor, J. Desrosiers, and J. M. V. Carvalho, "Dual-optimal inequalities for stabilized column generation," *Operations Research*, vol. 54, no. 3, pp. 454–463, 2006.

[13] J. M. V. de Carvalho, "Using extra dual cuts to accelerate convergence in column generation," *INFORMS Journal on Computing*, vol. 17, no. 2, pp. 175–182, 2005.

[14] H. B. Amor and J. Desrosiers, "A proximal trust-region algorithm for column generation stabilization," *Computers & Operations Research*, vol. 33, pp. 910–927, 2006.

[15] L.-M. Rousseau, M. Gendreau, and D. Feillet, "Interior point stabilization for column generation," *Operations Research Letters*, vol. 35, no. 5, pp. 660–668, 2007.

[16] M. Leitner, G. R. Raidl, and U. Pferschy, "Accelerating column generation for a survivable network design problem," in *Proceedings of the International Network Optimization Conference 2009*, M. G. Scutellà *et al.*, Eds., Pisa, Italy, 2009.

[17] M. Leitner and G. R. Raidl, "Strong lower bounds for a survivable network design problem," in *ISCO 2010*, ser. Electronic Notes in Discrete Mathematics, M. Haouari and A. R. Mahjoub, Eds., vol. 36. Elsevier, 2010, pp. 295–302.

[18] M. Leitner, G. R. Raidl, and U. Pferschy, "Branch-and-price for a survivable network design problem," Vienna University of Technology, Vienna, Austria, Tech. Rep. TR 186–1–10–02, 2010, submitted to Networks.

[19] H. B. Amor, J. Desrosiers, and A. Frangioni, "On the choice of explicit stabilizing terms in column generation," *Discrete Applied Mathematics*, vol. 157, pp. 1167–1184, 2009.