A MODELING FRAMEWORK FOR THE ORDERED WEIGHTED AVERAGE SPANNING TREE PROGLEM

Elena Fernández
Statistics and Operations Research Department. Barcelona Tech
Campus Nord, C5-208. Jordi Girona 1-3. 08034 Barcelona
e.fernandez@upc.edu

Miguel Angel Pozo
Statistics and Operations Research Department. University of Seville
Tarfia s.n. 41012 Seville
miguelpozo@us.es

Justo Puerto
Statistics and Operations Research Department. University of Seville
Tarfia s.n. 41012 Seville
puerto@us.es

ABSTRACT

We study the multiobjective spanning tree problem Spanning Tree Problem from a modeling point of view. In particular, we use the ordered median objective function as an averaging operator to aggregate the vector of objective values of feasible solutions. This leads to the Ordered Weighted Average Spanning Tree Problem, which we study in this work. We use elimination tests and valid inequalities to reinforce an existing mathematical programming formulation, and we present a new formulation which reduces the number of decision variables. Preliminary computational experiments indicate that the new formulation reinforced with appropriate constraints can be effective for efficiently solving medium size instances.

KEYWORDS. Spanning trees. Multiobjective optimization. Ordered median.

Main area: Combinatorial Optimization

1. Introduction

Multiobjective combinatorial optimization deals with problems considering more than one viewpoint or scenario. They inherit the complexity difficulty of their scalar counterparts but in addition incorporate a new facet that comes from dealing with partial orders in the objective function space. The standard solution concept is the set of Pareto solutions. However, the number of Pareto solutions can grow exponentially with the size of the instance and the number of objectives. A first approach to overcome this difficulty focuses on a specific subset of the Pareto set, such as, for instance, the supported Pareto solutions (see, for instance, Ehrgott, 2005). Those are the Pareto solutions that optimize linear scalarizations of the different objectives. However, it is possible to exhibit instances for which even the number of supported solutions grows exponentially on the size of the instance. Furthermore, focusing on supported Pareto solutions a priori excludes compromise solutions that could be preferred by the decision maker. For this reason, more involved decision criteria have been proposed in the field of multicriteria decision making (Perny and Spanjaard, 2003). These include objectives focusing on one particular compromise solution, which, for tractability and decision theoretic reasons, seem to be a better approach when an aggregation function is available.

In some cases, a particularly important Pareto solution related to a weighted ordered average aggregating function is sought. Provided the required preferential information is available, and provided the objectives are comparable, an averaging operator can be used to aggregate the vector of objective values of feasible solutions. The ordered median objective function is very useful in this context since it assigns importance weights not to specific objectives but to their best and worst evaluations (sorted values). Ordered median objectives have been successfully used for addressing various types of combinatorial problems (see, for instance, Ogryczak and Tamir, 2003; Nickel and Puerto, 2005; or, Boland et al., 2006).

In this paper, we investigate the multiobjective version of the optimal spanning tree problem. Multiobjective spanning tree problems have already been studied by some authors, mostly for the biobjective case (see Hamacher and Ruhe, 1994; Andersen et al., 1996; Sourd and Spanjaard, 2008; Steiner and Radzik, 2008). Recently Galand and Perny (2007) approached the problem using Choquet optimization and Galand and Spanjaard (2012) present a first ordered median integer programming formulation to obtain compromise solutions of the multiobjective spanning tree problem. Our approach elaborates on this line and introduces reinforcements that enhance that formulation in the form of preprocessing, elimination tests and valid inequalities. In addition, we develop a new formulation that, in our computational experiments, outperforms the previous one and that can also be efficiently reinforced with different families of valid inequalities. The paper is structured as follows. Section 2 formally defines the problem that we address. In section 3 we present the first formulation, give some valid inequalities and study some elimination tests. Section 4 presents the second formulation which uses a smaller set of decision variables. Section 5 describes the computational experiments and presents the obtained results. The paper end in Section 6 with some comments and lines for future research.

2. The Ordered Weighted Average Optimization

Let $G=(V, E)$ be an undirected connected graph and let $c^i: E \to \mathbb{R}$, $i \in P = \{1 \ldots p\}$ denote p real cost functions defined over the edge set E. Let also ω_i, $i \in P$ denote a set of weights for the cost functions. Feasible solutions to the Ordered Weighted Average Spanning Tree Problem (OWASTP) are spanning trees on G. For a given spanning tree T,

$v^i(T) = \sum_{(u,v)\in T} c^i_{uv}$ denotes its cost with respect to objective $i\in P$. The value of T is a weighted average of the cost of T with respect to each of the p objectives, in which the weight of each cost $v^i(T)$, $i\in P$, depends on its order relative to the other costs. In particular, ω_1 is the weight for the largest cost, ω_2 the weight for the second-largest cost, and ω_p is the weight for the smallest cost. Thus, in order to evaluate T, the costs $v^i(T)$, $i\in P$ have to de ordered by non-increasing values. Let π denote a permutation of the indices of P such that $v^{\pi(1)}(T)\geq v^{\pi(2)}(T)\geq \ldots v^{\pi(p)}(T)$. Then the value of T is

$$v(T) = \sum_{i\in P} \omega_i v^{\pi(i)}(T)$$

The Ordered Weighted Average Spanning Tree Problem (OWASTP) is to find a spanning tree in G of minimum total value.

For the sake of completeness, we start by describing a first linear integer programing formulation for the OWASTP that has been recently introduced by Galand and Spanjaard (2012). It uses several sets of decision variables. Some variables are used to model the order of the p cost functions ranked by non-increasing values. Other variables are used to model the structure of the combinatorial object (spanning tree in this case). For modeling the ordering, for all $i, j\in P$, y_{ij} indicates the value of cost function i if it occupies the j-th position in the ordering. In addition,

$$z_{ij} = \begin{cases} 1 & \text{if cost function } i \text{ occupies position } j \text{ in the ordering} \\ 0 & \text{otherwise} \end{cases}$$

For modeling the spanning tree we use a flow-based formulation, in which binary design variables x are related to continuous flow variables φ. In particular, for each $e=(u,v)\in E$ let

$$x_e = x_{uv} = \begin{cases} 1 & \text{if edge } (u,v) \text{ is in the spanning tree} \\ 0 & \text{otherwise} \end{cases}$$

For the flow variables we consider a directed network, with set of vertices V and set of arcs A which contains two arcs, one in each direction, associated with each edge of E. For each $(u,v)\in A$ we define the decision variables φ_{uv} which represents the amount of flow through arc (u, v). We use u_0 to denote an arbitrarily selected vertex. Then a formulation for the OWASTP is:

$$\min \quad \sum_{j\in P}\omega_j \sum_{i\in P} y_{ij} \qquad (1)$$

$$\sum_{j\in P} z_{ij} = 1 \qquad i\in P \qquad (2)$$

$$\sum_{i\in P} z_{ij} = 1 \qquad j\in P \qquad (3)$$

$$y_{ij} \le \left(\sum_{e\in E} c_e^i\right) z_{ij} \qquad i,j\in P \qquad (4)$$

$$\sum_{i\in P} y_{ij} \ge \sum_{i\in P} y_{i,j+1} \qquad j\in P\setminus\{p\} \qquad (5)$$

$$\sum_{i\in P} y_{ij} = \sum_{e\in E} c_e^i x_e \qquad i\in P \qquad (6)$$

$$\sum_{(u,v)\in A}\varphi_{uv} - \sum_{(v,u)\in A}\varphi_{vu} = n-1, \qquad u=u_0 \qquad (7)$$

$$\sum_{(u,v)\in A}\varphi_{uv} - \sum_{(v,u)\in A}\varphi_{vu} = -1, \qquad u\in V\setminus\{u_0\} \qquad (8)$$

$$\varphi_{uv} + \varphi_{vu} \le (n-1)x_{uv} \qquad (u,v)\in E \qquad (9)$$

$$\sum_{(u,v)\in E} x_{uv} = n-1 \qquad (10)$$

$$y_{ij} \ge 0, i,j\in P;\ \varphi_{uv}\ge 0, (u,v)\in A \qquad (11)$$

$$z_{ij}\in\{0,1\}, i,j\in P;\ x_e\in\{0,1\}, e\in E \qquad (12)$$

Constraints (2)-(3) define a permutation of the cost functions and (4) relate the y and z variables, to ensure that if y_{ij} takes a positive value, then the position of cost i is j. The condition that the permutation is the result of a non-increasing ordering of the cost functions values is guaranteed by constraints (5). Constraint (6) sets the cost of the spanning tree for objective i and assigns it to y_{ij} when cost i occupies position j. Constraints (7)-(10) define a spanning tree. In particular, (7)-(8) guarantee that one unit of flow arrives from the source vertex u_0 to any other vertex. Their main role is to guarantee that the graph induced by the arcs through which the flow circulates is connected and all vertices are "covered". Constraints (9) extend these two properties to the graph induced by the x variables, by imposing that all the edges used for sending flow in some direction are activated, whereas by constraint (10) the x variables define a spanning tree, since n-1 edges are selected.

3. Reinforced formulation for the OWASTP

As explained above, formulation (1)-(12) produces an optimal solution to the OWASTP. However, the difficulty of the OWASTP becomes evident already for relatively small size graphs with 30 nodes and 4 cost functions, for which more than 3,600 seconds of CPU time may be required by a commercial solver to find a provably optimal solution. Thus, reinforcing the above formulation may be crucial in order to be able to solve larger instances. The reinforced formulations will have the same set of feasible solutions as formulation (1)-(12), although they will have tighter domains for their linear programming (LP) relaxations, thus yielding tighter LP bounds.

Reinforcements can be attained by adding valid inequalities, by tightening some of the constraints in the formulation and by applying elimination tests for fixing variables. Next we propose reinforcements to formulation (1)-(12) based on each of these issues. First we include some additional notation.

For a given subset of vertices $S\subset V$, E(S) denotes the subset of edges of E with both end-vertices in S, and $\delta(S)=(S: V\backslash S)=\{e\in E : e=(u, v), u\in S, v\in V\backslash S\}$ denotes the edges in the cut-set between S and V\S. For a singleton we simply write $\delta(u)=\delta(\{u\})$. We use the compact notation $f(A)=\sum_{e\in A} f_e$ when $A\subseteq E$, and f is a vector or a function defined on E.

3.1 Valid inequalities

Next we present some valid inequalities we use to reinforce formulation (1)-(12).

- *Degree inequalities*. The flow variables φ and their relation to the design x variables, guarantee that the support graph induced by the x variables is connected, even for fractional values of the x variables. However, the well-known cut-set inequalities (Nemhauser and Wolsey, 1999) which must be satisfied by the x variables when they define a spanning tree, are not necessarily implied by constraints (7)-(9). In particular, for a given set of vertices $S\subset V$, the cut-set inequalities, which guarantee the connection of any subset of vertices with is complement, are:

$$x(\delta(S))\geq 1 \qquad (13)$$

Since there is a number of cut-set inequalities (13) which is exponential on $|V|$, we only consider the subset of such constraints associated with singletons. They are:

$$x(\delta(u))\geq 1 \quad u\in V \qquad (14)$$

We call them degree inequalities, since they impose that the degree of each vertex is at least one.

- *Subtour elimination constraints*. Another characteristic of spanning trees is that they contain no circuit. However, for fractional x values, this property is not implied by constraints (7), (8) and (9). Thus, the well-known subtour elimination constraints are valid for the OWASTP, and may reinforce the LP relaxation of formulation (1)-(12). They are:

$$x(E(S))\leq |S|-1 \setminus S\subset V \qquad (15)$$

The number of subtour elimination constraints is also exponential on $|V|$. Thus we only consider a (small) subset of such constraints associated with sets with three vertices. In particular, when $S=\{u_1, u_2, u_3\}$ with $u_1< u_2< u_3$ and (u_1, u_2), (u_2, u_3), $(u_1, u_3)\in E$ we obtain

$$x_{u_1u_2}+x_{u_2u_3}+x_{u_1u_3}\leq 2 \qquad (16)$$

- Constraints related to cost functions values. Let L_i and U_i denote the value of the minimum and maximum spanning tree relative to cost function $i\in P$, respectively. It is clear that L_i (U_i) are valid lower (upper) bounds on the value of objective i, independently of the position of cost function i in the ordering. Therefore we obtain the following two sets of constraints which are valid for the OWASTP.

$$L_i\leq \sum_{j\in P} y_{ij}\leq U_i \qquad i\in P \qquad (17)$$

- We can also establish a lower bound on the value of cost function $i\in P$ if it is ordered in position $j\in P$ by relating the x, y and z variables as follow:

$$y_{ij}\geq \sum_{e\in E} c_e^i x_e - U_i\left(1-z_{ij}\right) \qquad i, j\in P \qquad (18)$$

Observe that the above constraint imposes a lower bound on the value y_{ij} only when cost function $i \in P$ is ordered in position $j \in P$, and becomes inactive otherwise.

- We can also relate the values of two different cost functions between them, depending on their positions. In particular, for $i_1, i_2 \in P$, $i_1 \neq i_2$, $j \in P$ then,

$$\sum_{k=j+1}^{p} y_{i_1 k} \leq y_{i_2 j} + U_{i_1}\left(1 - z_{i_2 j} - z_{i_1 j}\right) \qquad (19)$$

Constraint (19) establishes that when cost function i_2 occupies position *j*, its value cannot be smaller than that of cost function i_1, provided that cost function i_1 is ordered after *j*. Observe that the constraint becomes inactive when i_1 is ordered before *j*, since in this case $\sum_{k=j+1}^{p} y_{i_1 k} = 0$, or when i_2 does not occupy position *j*.

- Let *U* be an upper bound on the value of any objective (for instance, $U = \max_{\{i \in P\}} u_i$. Then, for all $i,j \in P$

$$\sum_{k \in P} y_{k,j+1} \leq y_{ij} + U\left(1 - z_{ij}\right) \qquad (20)$$

Constraint (20) is only activated when objective *i* occupies position *j*. In this case, the value of the objective in position *j+1* cannot exceed the value of objective *i*.

In the following we denote *extended formulation* to formulation (1)-(12) reinforced with constraints (13)-(20).

3.2 Lower and upper bounds: Elimination tests

Several of the inequalities presented above use valid lower and upper bounds on the values of the different cost functions, L_i and U_i, respectively. As mentioned, the minimum and maximum spanning tree with respect to each cost function easily provide such bounds. However, tighter bounds can be very useful for obtaining tighter constraints. One possibility is to use lower and upper bounds on the value of each objective for the different positions in the ordering. If L_{ij} and U_{ij} respectively denote valid lower and upper bounds on the value of objective *i* if it occupies position j, then lower and upper bounds on the value of objective *i* are $L_i = \min_{\{j \in P\}} L_{ij}$ and $U_i = \max_{\{j \in P\}} U_{ij}$, respectively. For $i, j \in$ P given, L_{ij} and U_{ij} can be obtained in different ways. One possibility is to solve the linear programming (LP) relaxation of the extended formulation, for the minimization and the maximization of cost function *i*, with the additional constraint that it occupies position *j*. In this case L_{ij} (U_{ij}) is the optimal value of the minimization (maximization) of objective $\sum_{e \in E} c_e^i x_e$ subject to constraints (2)-(20), in which we fix the ordering variable at value 1, i.e. z_{ij}=1.

Next we present simple tests which can help to eliminate some variables by fixing their values. Broadly speaking these tests compare the value of a lower bound associated with the decision of setting (or not setting) objective *i* at position *j* with the value of a known upper bound. If the value of the lower bound exceeds the value of the upper bound, the associated decision variable can be fixed. Any spanning tree yields a valid upper bound, which corresponds to its value with respect to the objective function (1). In the following we use *U* to denote the value of the upper bound corresponding to the best-know solution. We also denote by L_{ij}^0 to the optimal value of the minimization of objective $\sum_{e \in E} c_e^i x_e$ subject to constraints (2)-(20) in which we fix the ordering variable at value 0, i.e. z_{ij} =0. Then for each $i \in P$, $j \in P$ we have

- If $L_{ij} > U$ then z_{ij} =0 (no optimal solution will have objective *i* in position *j*).

- If L_{ij}^0 >U then z_{ij} =1 (no optimal solution will not have objective *i* in position *j*).

4. Alternative formulation for the OWASTP with one index sorting variables

In this section we present an alternative formulation for the OWASTP. The main difference is that we no longer use two index decision variables to represent the value of objective functions depending on their sorted position. In particular, we now use two new sets of variables y_i , θ_i, $i\in P$ defined as:

$$y_i = \text{Value of the } i\text{-th objective fuction}$$

$$\vartheta_i = \text{Value of the objective fuction sorted in position } i$$

$$\min \quad \sum_{j\in P}\omega_j\theta_j \qquad (1')$$

$$\sum_{j\in P} z_{ij} = 1 \qquad i\in P \qquad (2')$$

$$\sum_{i\in P} z_{ij} = 1 \qquad j\in P \qquad (3')$$

$$y_i = \sum_{e\in E} c_e^i x_e \qquad i\in P \qquad (4')$$

$$\theta_j \ge y_i - U_i\left(\sum_{k=j}^{p} z_{ik}\right) \qquad i,j\in P \qquad (5')$$

$$\theta_j \ge \theta_{j+1} \qquad j\in P\setminus\{p\} \qquad (6')$$

$$\sum_{(u,v)\in A}\varphi_{uv} - \sum_{(v,u)\in A}\varphi_{vu} = n-1 \qquad u=u_0 \qquad (7')$$

$$\sum_{(u,v)\in A}\varphi_{uv} - \sum_{(v,u)\in A}\varphi_{vu} = -1 \qquad u\in V\setminus\{u_0\} \qquad (8')$$

$$\varphi_{uv} + \varphi_{vu} \le (n-1)x_{uv} \qquad (u,v)\in E \qquad (9')$$

$$\sum_{(u,v)\in E} x_{uv} = n-1 \qquad (10')$$

$$y_i \ge 0, i\in P;\ \varphi_{uv} \ge 0, (u,v)\in A \qquad (11')$$

$$z_{ij} \in \{0,1\}, i,j\in P;\ x_e \in \{0,1\}, e\in E \qquad (12')$$

Constraints (2')-(3') define a permutation of the cost functions and (4') define the objective value *y*. Original objective values and positions of sorted objectives are related in constraints (5'). By constraints (6') the permutation is the result of a non-increasing ordering of the cost functions values. Constraints (7')–(10') define a spanning tree. In particular, (7')-(8') guarantee that one unit of flow arrives from the source vertex u_0 to any other vertex. Their main role is to guarantee that the graph induced by the arcs through which the flow circulates is connected and all vertices are "covered'". Constraints (9') extend these two properties to the graph induced by the *x* variables, by imposing that all the edges used for sending flow in some direction are activated, whereas by constraint (10') the *x* variables define a spanning tree, since n-1 edges are selected.

4.1 Reinforcement of the formulation

In addition to all the valid inequalities developed in Section 3.2, we can add some new valid inequalities to this formulation. First of all, one can reinforce the lower bound of the sorted objective function in position *j* by using stronger upper bounds, UB_{ij}, on the value of the objective function *i* in the *j*-th position. These inequalities are related to (5').

$$\theta_j \geq y_i - U_{ij}(1 - z_{ik}) \qquad i, j \in P \tag{21}$$

Next, we observe that for any subset $I \subseteq P$, of size k=1, … ,p

$$\sum_{j=1}^{k} \theta_j \geq \sum_{j \in I} y_i \tag{22}$$

In particular, it is very effective the case I=P, giving rise to

$$\sum_{j \in P} \theta_j \geq \sum_{j \in P} y_i \tag{23}$$

There are also different lower bounds of the objective value sorted in position j:

$$\theta_j \geq \sum_{i \in P} L_{ij} z_{ij} \qquad j \in P \tag{24}$$

5. Computational experiments

In this section we report on the results of some computational experiments we have run, in order to compare empirically the proposed formulations. In these experiments we study a particular case of the OWASTP operator, namely the Hurwicz criterion, defined as $\alpha.\max_{i \in P} y_i + (1- \alpha).\min_{i \in P} y_i$. The test instances are defined as follows. All considered graphs are complete with $|V|$ ranging from 20 to 60, and a number of objectives of 3 or 5. The components of the cost vectors are randomly drawn from a uniform distribution on [1,100]. The considered values of α are {0.4, 0.6, 0.8}. For each selection of the parameters (|V|, |P|, α), 10 instances were randomly generated so, in total, we have a set of 300 benchmark instances. All instances were solved with the MIP Xpress optimizer, under a Windows 7 environment in an Intel(R) Core(TM)i7 CPU 2.93 GHz processor and 8 GB RAM. Default values were used for all parameters. A CPU time limit of 750 seconds was set.

Table 1 presents the results for formulation (1)-(12) without any reinforcement. Each row corresponds to a group of 10 instances with the same characteristics (|V|, |P|, α). The first two columns indicate the number of objectives and the number of vertices, respectively. Column under α gives the value in consideration for the parameter of the Hurwicz criterion. Column marked with (#) indicates the number of instances that could be solved within the maximum CPU time limit. The entries on the last three columns give information on the CPU time needed to optimally solve the instances. In particular s_{min} gives the smallest CPU time over all the instances of the group, s_{max} the biggest CPU time over the 10 instances of the group and $\bar{s}$ the average CPU time of the instances in the group. All times are measured in seconds. Mean values have been computed only over those instances solved in less than 750s. Whenever the time limit of 750s is reached for the 10 instances in the group, the entry is marked with (-). Tables 2 and 3 report analogous results for formulation (1)-(12) reinforced with constraints (13)-(20), and for formulation (1')-(12') reinforced with constraints (21)-(24), respectively.

By observing these tables, we conclude that reinforcement (13)-(20) improves considerably formulation (1)-(12) but, as could be expected, the best results are obtained by using (1')-(12') reinforced with constraints (21)-(24). Observe that, for a fixed set of parameters (|V|, |P|, α), the total number of instances optimally solved is not necessarily the same in the different tables. Thus, the averages are (possibly) computed with different number of instances. For this reason we can observe that some average running times of Table 3 are higher than the corresponding values in Table 2 (e.g. row |P|=3, |V|=60, α =0.6). Additionally, it seems that parameter α impacts on the running times increasing them according to the following sequence of values α =0.6, 0.4, 0.8.

6. Conclusions

In this work we have presented reinforced mathematical programming formulations for the OWASTP as well as a new formulation which reduces the number of decision variables. This new formulation reinforced with appropriate constraints has shown to be very promising for efficiently solving many medium size OWASTP instances. However, from the obtained results it is also clear that for solving larger OWASTP instances with more objective functions further improvements are needed. Our current research focuses on the design of more sophisticated elimination tests as well as from alternative formulations leading to tighter LP bounds.

7. Acknowledgements

This research has been partially supported by the Spanish Ministry of Science and Education through grants MTM2009-14039-C06-05 and MTM2010-19576-C02-01, by Junta de Andalucía grant FQM5849 and by ERDF funds. This support is gratefully acknowledged.

8. References

Andersen, K.A., Jörnsten, K., Lind, M. (1996), On bicriterion minimal spanning trees: an approximation. *Computers & Operations Research*, 23 (12), 1171-1182.

Boland, N., Domínguez-Marín, P., Nickel, S., Puerto, J. (2006), Exact procedures for solving the discrete ordered median problem. *Computers & Operations Research*, 33 (11) 3270-3300

Ehrgott, M., *Multicriteria optimization* (2nd ed.), Springer, Heidelberg, Germany, 2005.

Galand, L., Perny, P. (2007) Search for Choquet-optimal paths under uncertainty. *Proceedings of the 23rd conference on uncertainty in artificial intelligence*, AAAI Press, Vancouver, Canada, 125-132.

Galand, L., Spanjaard, O., (2012) Exact algorithms for OWA-optimization in multiobjective spanning tree problems *Computers & Operations Research* 39 (2012) 1540–1554.

Hamacher, H.W., Ruhe, G. (1994), On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52, 209-230.

Nemhauser G.L., Wolsey L., Integer and Combinatorial Optimization, Wiley-Interscience, New York, 1999.

Nickel, S., Puerto, J., *Location theory: a unified approach*, Springer-Verlag, Heidelberg, Germany, 2005.

Ogryczak, W. Tamir, A. (2003), Minimizing the sum of the *k* largest functions in linear time. *Information Processing Letters*, 85 (3), 117-122.

Perny, P., Spanjaard, O. (2003), An axiomatic approach to robustness in search problems with multiple scenarios. *Proceedings of the 19th conference on uncertainty in artificial intelligence,* 469-476.

Sourd, F., Spanjaard, O. (2008), A multi-objective branch-and-bound framework. Application to the bi-objective spanning tree problem. *INFORMS Journal of Computing*, 20 (3), 472-484.

Steiner, S., Radzik, T. (2008), Computing all efficient solutions of the biobjective minimum spanning tree problem. *Computers & Operations Research*, 35 (1) 198-211.

|P|	|V|	α	#	s_{min}	$\bar{s}$	s_{max}
3	20	0.4	10	0.53	0.76	1.26
3	20	0.6	10	0.47	0.65	0.89
3	20	0.8	10	0.58	2.18	8.13
3	30	0.4	10	1.15	13.41	61.89
3	30	0.6	10	0.68	2.36	6.3
3	30	0.8	10	0.64	15.4	58.42
3	40	0.4	8	3.42	36.99	-
3	40	0.6	9	2.73	16.83	-
3	40	0.8	9	3.86	59.72	-
3	50	0.4	7	9.85	331.69	-
3	50	0.6	8	4.44	75.92	-
3	50	0.8	8	6.63	124.26	-
3	60	0.4	5	29.18	98.45	-
3	60	0.6	5	13.53	127.33	-
3	60	0.8	2	19.92	150.35	-
5	20	0.4	10	1.53	6.49	24.08
5	20	0.6	10	5.32	21.53	73.8
5	20	0.8	10	3.03	25.97	51.15
5	30	0.4	9	8.45	100.41	-
5	30	0.6	7	12.36	114.3	-
5	30	0.8	6	96.24	270.63	-
5	40	0.4	2	93.19	349.12	-
5	40	0.6	2	451.62	524.82	-
5	40	0.8	4	174.97	330.01	-
5	50	0.4	0	-	-	-
5	50	0.6	0	-	-	-
5	50	0.8	0	-	-	-
5	60	0.4	0	-	-	-
5	60	0.6	0	-	-	-
5	60	0.8	0	-	-	-

Table 1: Results for formulation (1)-(12).

|P|	|V|	α	#	s_{min}	$\bar{s}$	s_{max}
3	20	0.4	10	0.78	0.98	1.34
3	20	0.6	10	0.64	0.79	0.95
3	20	0.8	10	0.52	1.86	9.02
3	30	0.4	10	1.87	3.44	8.72
3	30	0.6	10	1.15	1.87	3.32
3	30	0.8	10	1.01	26.54	155.21
3	40	0.4	10	4.15	77.71	643.25
3	40	0.6	10	2.34	9.8	37.46
3	40	0.8	9	1.36	30.75	-
3	50	0.4	9	14.85	149.12	-
3	50	0.6	10	5.84	96.53	353.78
3	50	0.8	9	4.57	130.7	-
3	60	0.4	8	30.75	150.49	-
3	60	0.6	6	26.69	113.49	-
3	60	0.8	4	13.09	278.43	-
5	20	0.4	10	1.36	3.18	11.53
5	20	0.6	10	3.93	9.96	25.9
5	20	0.8	10	1.78	22.45	73.34
5	30	0.4	9	6.18	74.6	-
5	30	0.6	9	8.46	140.55	-
5	30	0.8	7	37.53	216.93	-
5	40	0.4	6	59.37	298.2	-
5	40	0.6	5	37.74	239.35	-
5	40	0.8	3	99.23	267.88	-
5	50	0.4	5	224.52	403.99	-
5	50	0.6	1	137.91	137.91	-
5	50	0.8	1	440.01	440.01	-
5	60	0.4	0	-	-	-
5	60	0.6	1	535.07	535.07	-
5	60	0.8	1	400.89	400.89	-

Table 2: Results for formulation (1)-(12) reinforced with inequalities (13)-(20).

|P	|V|	α	#	s_{min}	$\overline{s}$	s_{max}
3	20	0.4	10	0.48	0.65	0.89
3	20	0.6	10	0.45	0.57	0.69
3	20	0.8	10	0.47	1	2.89
3	30	0.4	10	0.56	2.3	4.85
3	30	0.6	10	0.78	1.54	3.59
3	30	0.8	10	0.86	4.52	11.79
3	40	0.4	10	2.28	53.75	416.93
3	40	0.6	10	2.04	7.72	17.3
3	40	0.8	10	1.7	20.24	40.25
3	50	0.4	9	4.99	164.34	-
3	50	0.6	10	3.12	33.59	145.49
3	50	0.8	9	2.04	18.13	-
3	60	0.4	8	17.91	166.88	-
3	60	0.6	7	19.95	172.57	-
3	60	0.8	6	11	240.13	-
5	20	0.4	10	0.48	0.85	1.03
5	20	0.6	10	0.72	1.25	2.03
5	20	0.8	10	0.58	0.97	1.86
5	30	0.4	10	1.26	8.66	24.66
5	30	0.6	9	1.62	60.04	-
5	30	0.8	10	2.33	38.69	244.27
5	40	0.4	8	7.11	47.34	-
5	40	0.6	5	30.06	249.13	-
5	40	0.8	8	5.12	131.39	-
5	50	0.4	7	22.32	220.37	-
5	50	0.6	6	22.39	180.83	-
5	50	0.8	5	144	295.8	-
5	60	0.4	2	118.44	211.15	-
5	60	0.6	2	202.18	318.4	-

Table 2: Results for formulation (1’)-(12’) reinforced with inequalities (13)-(20).