An Integer Linear Programming Solution to the Telescope Network Scheduling Problem

Sotiria Lampoudi[1,*], Eric Saunders[2] and Jason Eastman[2]
[1]Earth Research Institute, 6832 Ellison Hall, University of California, Santa Barbara, California 93106-3060, U.S.A.
[2]Las Cumbres Observatory Global Telescope Network, 6740 Cortona Dr., Suite 102, Goleta, California 93117, U.S.A.

Keywords: Scheduling, Integer Linear Programming, Astronomy.

Abstract: Telescope networks are gaining traction due to their promise of higher resource utilization than single telescopes and as enablers of novel astronomical observation modes. However, as telescope network sizes increase, the possibility of scheduling them completely or even semi-manually disappears. In an earlier paper, a step towards software telescope scheduling was made with the specification of the Reservation formalism, through the use of which astronomers can express their complex observation needs and preferences. In this paper we build on that work. We present a solution to the discretized version of the problem of scheduling a telescope network. We derive a solvable integer linear programming (ILP) model based on the Reservation formalism. We show computational results verifying its correctness, and confirm that our Gurobi-based implementation can address problems of realistic size. Finally, we extend the ILP model to also handle the novel observation requests that can be specified using the more advanced Compound Reservation formalism.

1 INTRODUCTION

Telescope networks have the potential to enable increased resource utilization and novel observation modalities. Historically, observation requests for single telescopes were made in human-readable form, and any conflicts between requests were resolved by a person, often working directly with the requesting astronomer in a tight feedback loop. This type of manual scheduling is not practical for general-purpose telescope networks containing more than a small number of telescopes, due to the large number of competing requests received for a typical scheduling interval, and the added complexity of choosing among multiple resources. Further, in networks wishing to enable the study of fast transient phenomena, manual scheduling is infeasible due to the need for near-real-time re-scheduling responsiveness, which is necessary to achieve these scientific objectives.

Las Cumbres Observatory Global Telescope (LCOGT) is a robotic telescope network that currently (in September 2014) includes two 2m and nine 1m telescopes, with plans to add a number of 0.4m telescopes in the near future (Brown et al., 2013). The telescopes are robotically controlled and connected via the Internet to LCOGT headquarters in California. Professional astronomers, citizen scientists and educators can apply for access to the network on a biannual semester basis. The scientific merit of proposals is assessed by a Time Allocation Committee (TAC), which assigns each accepted project some amount of total time on the network and a scalar per-unit-time priority. Each project then requests specific astronomical observations to be conducted, not exceeding the project's total time allocation. The motivation for the contributions of this paper is the design and deployment of a software telescope network scheduler for the LCOGT network (Saunders et al., 2014).

A software solution stack for scheduling a telescope network has three components: (a) methods allowing the network's users to make observation requests, (b) a scheduling algorithm capable of resolving conflicts between users' requests to produce a viable schedule, and (c) additional control logic that adds awareness of the state of the network, manages schedule re-calculation (due to new input, weather, network outages and other reasons), and deals with request completion. In (Lampoudi and Saunders, 2013) a formalism was developed for allowing astronomers to express their complex observation needs and preferences in an unambiguous, machine-readable way

*Author to whom correspondence should be addressed. Current Affiliation: Liquid Robotics Inc., 1329 Moffet Park Dr, Sunnyvale, California, 94089, USA

Lampoudi S., Saunders E. and Eastman J..
An Integer Linear Programming Solution to the Telescope Network Scheduling Problem.
DOI: 10.5220/0005207003310337
In *Proceedings of the International Conference on Operations Research and Enterprise Systems* (ICORES-2015), pages 331-337
ISBN: 978-989-758-075-8

that would allow software to arbitrate among them. The contribution of the present paper is a solvable integer linear programming (ILP) model for the offline, discretized version of the scheduling problem expressed by this formalism.

This paper is organized as follows: in section 2 we present the Reservation formalism that is used to communicate astronomers' requests to the scheduler. Section 3 presents the solvable ILP model of the scheduling problem. Section 4 presents computational results confirming the correctness and evaluating the performance of the ILP solution. Section 5 extends the model of section 3 to include the more complex Compound Reservations possible in telescope networks. Related and future work are discussed in the final section.

2 THE RESERVATION FORMALISM

A request for an observation on a telescope network contains two types of information: (a) observation-specific information about the target of the observation, which instrument (i.e. camera or spectrograph) to use, the exposure settings, etc, and (b) information about when, where and for how long the observation can occur, based on astronomical factors that can be calculated *a priori*, such as visibility of the target during local night, lack of interference by the moon, and so on. Information contained in (a) is necessary so that a robotic telescope or a human operator can carry out the observation. But it is not necessary for choosing which of many requests to fulfill, when and where. This task, the scheduling of the telescope network, is performed solely on the basis of the information contained in (b). In our work the information contained in (b) is encapsulated in a "Reservation": a representation of a request for exclusive access to a resource during one contiguous chunk of time at one or more specific times in the future.

As formally specified in (Lampoudi and Saunders, 2013), a Reservation R is a 4-tuple (d, p, t, W) where:

- d is a scalar duration,
- p is a scalar priority,
- t is a resource (i.e. telescope),
- W is a list of "windows of opportunity"

Windows of opportunity specify the times during which the observation is possible; that is, the entire observation must fit within a single window of opportunity.

In the vocabulary of a telescope network, a Reservation R is a request by a project with priority p for exclusive access of duration d to resource t during one of the windows in the list W.

Typically, however, an observation can be carried out on one of many telescopes. According to the above definition of Reservation, a request with multiple resources (and corresponding windows of opportunity for each resource) will result in multiple "or"-ed Reservations. Because this is such a common occurrence, for compactness and with no loss of generality, we simply extend the above definition to merge multi-resource Reservations into a single Reservation. That is, each Reservation is now permitted to include a list of resources, instead of a single resource, and windows of opportunity become subscripted by resource. The resulting definition of Reservation is the 4-tuple (d, p, T, W) where:

- d is a scalar duration,
- p is a scalar priority,
- T is a list of resources (i.e. telescopes), $t_i \in T$,
- $W_i \in W$ are lists of "windows of opportunity", where list W_i is the list of windows corresponding to t_i.

3 THE INTEGER LINEAR PROGRAMMING MODEL

Given a list of Reservations we wish to find a maximum priority subset, the subset of scheduled reservations, and an assignment of a specific resource and start time for each scheduled reservation, such that there are no overlaps between scheduled reservations.

This is an offline, multi-resource, interval scheduling problem with slack. The slack is introduced by the fact that windows can be longer than the duration of a Reservation.

Our ILP model is inspired by a similar approach to a problem in truck scheduling (Lee et al., 2012). We first discretize time into "slots", which can be of uniform or non-uniform lengths. Each slot is defined by the resource to which it corresponds and a (start time, duration) or (start time, end time) tuple that we abbreviate as (timeslice) in the text that follows. We then express the non-overlap constraints as linear inequalities. Boolean variables are used to select between the possible starting slots for each Reservation, and the sum of the priorities of scheduled Reservations is maximised.

The resulting ILP resembles a weighted maximum set packing problem, which is known to be NP-complete (Garey and Johnson, 1990).

Specifically, our model formulation is as follows:

3.1 Parameters

- I: set of reservations
- T: set of slots, each specified as a tuple: (resource, timeslice)
- S_i: set of possible start slots for reservation i
- $a_{ikt} = 1$ if starting reservation i at $k \in S_i$ means that it will occupy slot t; 0 otherwise
- p_i: priority of reservation i

3.2 Decision Variables

$Y_{ik} = 1$ if reservation i starts at $k \in S_i$; 0 otherwise

3.3 Objectives

Maximise the sum of the priorities of scheduled Reservations.

$$\max \sum_{i \in I} \sum_{k \in S_i} p_i Y_{ik} \tag{1}$$

3.4 Constraints

No reservation should be scheduled for more than one start.

$$\sum_{k \in S_i} Y_{ik} \leq 1, \forall i \in I \tag{2}$$

No more than one reservation should be scheduled in each slot.

$$\sum_{i \in I} \sum_{k \in S_i} a_{ikt} Y_{ik} \leq 1, \forall t \in T \tag{3}$$

Decision variable must be binary.

$$Y_{ik} \in \{0, 1\}, \forall i \in I, \forall k \in S_i \tag{4}$$

The inequality constraint matrix contains $|I| + |T|$ rows, i.e. the sum of the number of reservations and time slots.

4 COMPUTATIONAL RESULTS

4.1 Correctness

The ILP model we have just described is implemented in a software component called the “scheduling kernel”. This forms the core of the software stack responsible for managing the telescope network. Our current implementation of the kernel is in Python, and uses the Gurobi solver (Gurobi Optimization, 2014). The input to the kernel is a list of Reservation objects which are direct implementations of the concept of a Reservation. The kernel translates this list of Reservations into a model description that is passed to Gurobi, and invokes the solver. When the solver completes its run, the kernel uses the output to modify the Reservation objects to reflect whether or not they were scheduled, and, if they were, at what resource and starting time.

As is common software engineering practice, the kernel was unit tested using a variety of small scheduling scenarios that can be solved manually. But it was also desirable to validate that the kernel’s performance is what one would expect on a larger scale.

This type of validation can be achieved by using large scheduling scenarios for which the optimal scheduling outcome is known *a priori*, due to the way in which they were constructed. When these scenarios are run through the scheduling kernel, it is possible to compare the experimental performance of the kernel to this theoretically optimal and achievable outcome.

The “subscription rate” of the network is defined as the ratio of the total amount of time requested over the total amount of time available for scheduling. The total amount of time requested is the sum of the durations of all the reservations submitted to the scheduler. The total amount of time available for scheduling is the time covered by the union of all windows of opportunity.

The subscription rate is a property of the input to the scheduler. To characterize the outcome of a scheduling run we need a corresponding performance metric. This is the “scheduled/requested” (s/r) ratio, that is the ratio of time scheduled (the sum of the durations of all scheduled reservations) over the ratio of time requested (the sum of the durations of all reservations that were submitted).

For subscription rates below 100% (“undersubscribed”), it is possible to construct problem instances for which the optimal s/r ratio of 100% is achievable. Given those problem instances, a well-functioning scheduler should achieve s/r of nearly 100%.

For subscription rates above 100% (“oversubscribed”), it is possible to construct problem instances for which the optimal s/r ratio is known. Specifically, we constructed cases for which the optimal s/r was the inverse of the subscription rate – another way to express that is to say that utilization was 100%. On those problem instances a well-functioning scheduler should achieve close to this theoretically optimal s/r.

To produce experimental data that can be compared to the optimal values we conducted 15 simulations of a telescope network, spanning subscription rates between 10% and 150%. The size of the network was chosen to be nine telescopes, and the time slices on all telescopes were set to be 5 minutes long.

For each individual simulation run, i.e., for each subscription rate value, we generated an ensemble of hundreds of reservations for which we knew, by construction, that an optimal or close to optimal schedule was feasible. Then the scheduling problem was made artificially harder in two ways: first, all reservations were assigned the same 24 hour window of opportunity; second, all were also randomly assigned to be possible on additional resources. This had the effect of introducing large amounts of slack and seeming contention to the problem.

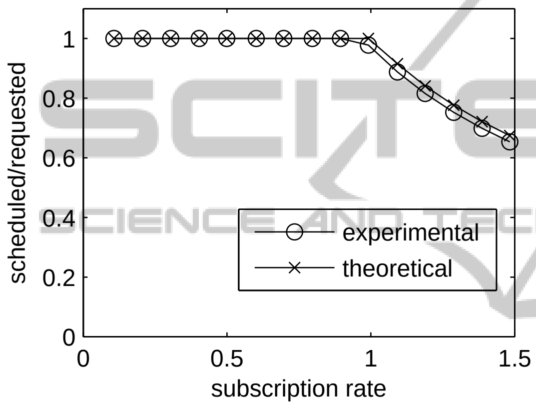

Figure 1: The scheduling kernel is able to achieve optimal performance in cases of undersubscription, and performs close to optimally in cases of oversubscription. The discrepancy is accounted for by the time wasted due to the discretization of time into slots.

Figure 1 shows that the kernel was able to match theoretically optimal performance very closely. The small mismatch that begins to occur around 100% subscription can be attributed to the small amount of time that is wasted due to the discretization of time into slots. In the case of the artificial scenarios used for this test, this wasted amount of time can be calculated.

In real-world scenarios the impact of that wasted time is an open problem. The amount of wasted time depends on (a) the distribution of Reservation durations, and (b) the choice of slot sizes. Clearly, smaller slots decrease the expected amount of wasted time, but increase the size of the ILP optimization problem. We plan to study this effect in simulation, by modeling the distribution of durations so that we can generate appropriate synthetic workloads, and empirically, by running the kernel on real inputs but, in "parallel", using different hypothetical slot sizes.

4.2 Performance

To give an idea of what currently constitutes typical and exceptional operating conditions for the LCOGT scheduling kernel, in this section we report timings from two real-world runs. The first is a randomly chosen typical recent run (date: 2014-08-29). The second is the largest run that occurred since the scheduler began official operations in May 2014 (date: 2014-06-21). They both occurred on the same server, a 16 core Intel Xeon L5530 2.4GHz, with 24GB of RAM. The implementation ran under Python 2.7.5 and Gurobi 5.6.2 on a Linux OS. Gurobi was configured to use 16 threads, i.e. all the available cores.

In the typical run the input included 833 possible Reservations, and seven of eleven resources were available for scheduling. This resulted in a problem description with 20,895 rows, 138,635 columns and 479,240 nonzeros, as reported by Gurobi. This was reduced to 8,826 rows, 92,093 columns, 293,297 nonzeros by the Gurobi pre-solver. 650 Reservations were ultimately scheduled. Wall clock time spent in the kernel (which includes the translation of the problem into a format that can serve as an input to Gurobi, a process that we have since further optimized) was 23.77 seconds; in the Gurobi pre-solver 6.56 seconds; in the Gurobi root relaxation routine 0.21 seconds; and in the Gurobi integer solver 11.50 seconds. In total, 23% of the time was spent in kernel overhead, with the remaining time spent in Gurobi.

The biggest scheduling run during the last few months had an input of 3864 possible Reservations, roughly four times as many as the typical run. The same fraction (7/11) of resources were available for scheduling at the time of this run. The final schedule included 2055 of the submitted Reservations. The total kernel wall time was 76.54 seconds, of which 17.22 seconds was spent in the Gurobi pre-solver; 20.39 seconds were spent in the Gurobi root relaxation step; and 24.83 seconds were spent in the Gurobi integer solver. The kernel overhead was thus 18% for this larger problem. These measurements are summarized in Table 1.

Table 1: Measurements for typical and largest runs.

	Typical	Largest
reservation count	833	3864
wall time (s)	23.77	76.54
% kernel overhead	23	18
pre-solver (s)	6.56	17.22
root relaxation (s)	0.21	20.39
integer solution (s)	11.5	24.83

5 EXTENSION TO COMPOUND RESERVATIONS

Astronomical observations requested by a particular project are usually part of a larger scientific programme, so they are frequently not independent of each other. When, as is most common, the interdependence between observations is in the targets, instruments, and exposures of observations, it does not affect scheduling. But it is occasionally helpful to provide a way to express a limited form of interdependence between the *scheduling status* of observations, i.e. whether or not they were scheduled. This is useful in situations where one of several alternative Reservations can fulfill the same scientific objective, or when sets of Reservations must be scheduled in an "all-or-none" fashion in order to be scientifically useful. We allow for this limited type of dependency between Reservations via the concept of a Compound Reservation, first introduced in (Lampoudi and Saunders, 2013).

A Compound Reservation is a set of Reservations, inter-connected by one of two logical operators: AND and ONE-OF.

The AND operator is the traditional conjunction operator. (r_1 AND r_2) means simply that either both reservations r_1 and r_2 should be scheduled, or neither should be scheduled. (r_1 AND r_2 AND ... AND r_i) is, by extension, defined as one would expect.

The ONE-OF operator is equivalent to a "one-hot circuit" in digital circuit design. (r_1 ONE-OF r_2) means that either reservation r_1 or r_2 should be scheduled, but not both. (Because of the no-overlap constraint, it is possible that neither reservation can be scheduled, making this a set packing, rather than a set partitioning constraint.) For two arguments ONE-OF is equivalent to XOR. The reason we use the notation ONE-OF rather than XOR is that the implementation of a greater-than-two input XOR is not unique. The most common implementation (i.e. wiring diagram) of XOR for greater than two arguments yields a parity checker. What we need is a circuit that evaluates to True when exactly one of its arguments is true. Since the term "one-hot" is not commonly used outside digital design, we use the more intuitive label "ONE-OF" for this operator.

Single-level Compound Reservations, as we describe them here, enable some of the novel capabilities of a telescope network. They make it possible to schedule an observation so that it occurs on one of multiple alternative resources requiring different exposure times (using ONE-OF), potentially increasing utilization by leveraging flexibility. Compound Reservations also make it possible to schedule time-series observations in an all-or-none fashion (using AND), decreasing time wasted obtaining partial data. They make it possible to conduct concurrent observations of a single target, or many correlated targets, from multiple resources (using AND), which previously required human coordination. Finally, on the LCOGT network, which is global, they enable the tracking of stationary or moving targets in spite of the earth's rotation, using a succession of resources (AND), which has never before been possible. Importantly, although an arbitrary level of Compound Reservation nesting is conceptually possible but computationally intractable, all these capabilities are gained using a single level of nesting, which it is feasible to schedule.

The presence of Compound Reservations modifies the problem definition as follows: Given a list of Reservations, where some are possibly grouped into Compound Reservations, we wish to find a maximum priority subset of scheduled reservations, and assign them specific resources and start times, such that there are no overlaps between scheduled reservations, and the constraints implied by the Compound Reservations (single-level ANDs and ONE-OFs) are satisfied.

This extension adds the following parameters to the list of section 3.1:

- O: the set of ONE-OF constraints
- A: the set of AND constraints

The following constraints are added to those of section 3.4:

ONE-OF constraints.

$$\sum_{i \in r} \sum_{k \in S_i} Y_{ik} \leq 1, \forall r \in O_j, O_j \in O \tag{5}$$

AND constraints.

$$\sum_{k \in S_i} Y_{ik} - \sum_{k \in S_j} Y_{jk} = 0, \forall i, j \in r, \forall r \in A_j, A_j \in A \tag{6}$$

The size of the inequality constraint matrix is modified to be $|I| + |T| + |O|$ rows, i.e. the sum of the number of reservations, time slots and ONE-OF constraints.

Finally, AND constraints introduce an equality constraint matrix with number of rows proportional to the number of reservations participating in AND constraints.

6 RELATED AND FUTURE WORK

The literature on ILP for interval scheduling is ubiquitous (see, e.g. (Schrijver, 1986), (Graham et al., 1979) and (Potts and Strusevich, 2009)). Our own effort to model the telescope network scheduling problem as an ILP problem was inspired by a similar (though more complicated, due to the presence of multiple objectives) model for truck scheduling (Lee et al., 2012).

Early work in telescope scheduling, which was surveyed in (Lampoudi and Saunders, 2013), was of a highly practical and heuristic nature. In general those early approaches were difficult to evaluate for fitness of purpose, and they commonly handled complexity, especially dynamic volatility, through direct human intervention and decision-making.

More recently, methods adopted from the Artificial Intelligence community, e.g. neural networks (Colomé et al., 2014), and from Operations Research, e.g. genetic algorithms in the context of optimization (Garcia-Piquer et al., 2014), have been making inroads in telescope network scheduling. A necessary shift is occuring in the field towards methods whose performance can be quantitatively compared to either theoretical models or simulation outcomes.

For completeness, it is worth noting that there have been two previous design iterations for the LCOGT scheduler. A randomised planning approach was proposed in (Brown and Baliber, 2007). In (Hawkins et al., 2010) the scheduling problem was broken into a hierarchy of seasonal, monthly and adaptive planning steps, but specific implementations for those steps were not proposed. Both of these were preliminary proposals, and were never fully implemented or evaluated. They both reflected a desire to steer clear of global optimisation, a stance that was justified by citing resource availability, volatility and computational cost. Given improvements in computational speeds, this stance was reversed, and our present optimization approach was adopted.

Our own work is divided between, on the one hand, efforts to gain a deeper understanding of the structure of the ILP optimization problem, (e.g. analysing the structure of the conflict graph) and, on the other hand, evaluating several practical questions concerning the scheduling kernel and its performance. One of these is the impact of the time discretization introduced by the ILP model, as we explained in section 4.2. Another is the choice of priority model, or more broadly, objective functions, in order to best match the science objectives of the network. Finally, the possibilities implied by the compound reservation scheduling capabilities of the kernel remain as yet uncharacterized. Such a complex feature requires both an excellent user interface to be useful, as well as considerable community training efforts to gain adoption. We anticipate that when these conditions come to fruition, new statistics will become available, and inform new models of telescope utilization, driving forward the next iteration of development and theoretical advances.

REFERENCES

Brown, T. M. and Baliber, N. (2007). Towards A Scheduler For The LCOGT Multi-telescope Network. In *American Astronomical Society Meeting Abstracts*, volume 39 of *Bulletin of the American Astronomical Society*, pages 807–807.

Brown, T. M., Baliber, N., Bianco, F. B., Bowman, M., Burleson, B., Conway, P., Crellin, M., Depagne, É., De Vera, J., Dilday, B., Dragomir, D., Dubberley, M., Eastman, J. D., Elphick, M., Falarski, M., Foale, S., Ford, M., Fulton, B. J., Garza, J., Gomez, E. L., Graham, M., Greene, R., Haldeman, B., Hawkins, E., Haworth, B., Haynes, R., Hidas, M., Hjelstrom, A. E., Howell, D. A., Hygelund, J., Lister, T. A., Lobdill, R., Martinez, J., Mullins, D. S., Norbury, M., Parrent, J., Paulson, R., Petry, D. L., Pickles, A., Posner, V., Rosing, W. E., Ross, R., Sand, D. J., Saunders, E. S., Shobbrook, J., Shporer, A., Street, R. A., Thomas, D., Tsapras, Y., Tufts, J. R., Valenti, S., Vander Horst, K., Walker, Z., White, G., and Willis, M. (2013). Las Cumbres Observatory Global Telescope Network. *PASP*, 125:1031–1055.

Colomé, J., Colomer, P., Campreciòs, J., Coiffard, T., de Oña, E., Pedaletti, G., Torres, D. F., and Garcia-Piquer, A. (2014). Artificial intelligence for the CTA Observatory scheduler . In *Proceedings of SPIE*, volume 9149, pages 91490H–91490H–15.

Garcia-Piquer, A., Guàrdia, J., Colomé, J., Ribas, I., Gesa, L., Morales, J. C., Pérez-Calpena, A., Seifert, W., Quirrenbach, A., Amado, P. J., Caballero, J. A., and Reiners, A. (2014). CARMENES instrument control system and operational scheduler . In *Proceedings of SPIE*, volume 9152, pages 915221–915221–15.

Garey, M. R. and Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Graham, R., Lawler, E., Lenstra, J., and Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In P.L. Hammer, E. J. and Korte, B., editors, *Discrete Optimization II Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium co-sponsored by IBM Canada and SIAM Banff, Aha. and Vancouver*, volume 5 of *Annals of Discrete Mathematics*, pages 287 – 326. Elsevier.

Gurobi Optimization, I. (2014). Gurobi optimizer reference manual.

Hawkins, E., Baliber, N., Bowman, M., Brown, T., Burleson, B., Foale, S., Ford, M., Lister, T., Norbury, M., Saunders, E., and Walker, Z. (2010). Scheduling observations on the LCOGT network. In Silva,

D. R., Peck, A. B., and Soifer, B. T., editors, *Observatory Operations: Strategies, Processes, and Systems III*, volume 7737. SPIE.

Lampoudi, S. and Saunders, E. (2013). Telescope network scheduling - rationale and formalisms. In Vitoriano, B. and Valente, F., editors, *ICORES'13*, pages 313–317. SciTePress.

Lee, S., Turner, J., Daskin, M. S., de Mello, T. H., and Smilowitz, K. (2012). Improving fleet utilization for carriers by interval scheduling. *European Journal of Operational Research*, 218(1):261–269.

Potts, C. N. and Strusevich, V. A. (2009). Fifty years of scheduling: a survey of milestones. *JORS*, 60(S1).

Saunders, E. S., Lampoudi, S., Lister, T. A., Norbury, M., and Walker, Z. (2014). Novel scheduling approaches in the era of multi-telescope networks. In *Proc. SPIE*, volume 9149, pages 91490E–91490E–7.

Schrijver, A. (1986). *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., New York, NY, USA.