Scheduling Telescope Observations for the European Southern Observatory

Michael Prümm
European Southern Observatory, Garching, Germany

Peter Nightingale
Department of Computer Science, University of York, UK

Felix Ulrich-Oltean
Department of Computer Science, University of York, UK

Abstract

The European Southern Observatory (ESO) provides state-of-the-art large telescope facilities at three sites in Chile, supported by 16 European member states. Astronomers submit proposals for sets of observations which are reviewed and ranked based on scientific merit, then a schedule is constructed respecting the ranking and aiming to make the fullest use of the various telescopes and numerous instruments. Currently a schedule covers six months, but in the near future ESO will switch to annual schedules.

Here we examine the most challenging scheduling problem encountered by ESO: scheduling the operations of the Very Large Telescope Interferometer (VLTI) on Paranal, Chile. Tasks to be scheduled include observations performed by ESO staff, 'visitor mode' periods where astronomers visit the site to use the telescopes, various maintenance tasks, and reconfiguration tasks taking multiple days. Typically a VLTI six-month schedule would contain approximately 450 activities. We explore global constraint models and a SAT encoding of the problem.

2012 ACM Subject Classification Theory of computation → Constraint and logic programming; Computing methodologies → Planning and scheduling

Keywords and phrases Modelling, Constraint Programming, Scheduling, SAT, Global Constraints

Digital Object Identifier 10.4230/LIPIcs.CP.2025.35

Category Short Paper

Supplementary Material Repository of models, instances and software for experiments
Models, data, software: `https://doi.org/10.15124/30f68e66-acf8-44a2-ae49-54b79381c498`

Funding *Peter Nightingale*: EPSRC grant EP/W001977/1
Felix Ulrich-Oltean: EPSRC grant EP/W001977/1

1 Introduction

The European Southern Observatory (ESO) provides a state-of-the-art facility at its site on Paranal, Chile, hosting a variety of instruments (see Figure 1). Astronomers submit proposals for sets of observations on instruments – these proposals are then ranked based on their scientific merit. A schedule is constructed to accommodate the observations from as many proposals as possible, respecting the ranking. This scheduling task is particularly challenging for the Very Large Telescope Interferometer (VLTI). The VLTI combines light from four 8.2 metre Unit Telescopes (UTs) and four movable 1.8 metre Auxiliary Telescopes (ATs) to create a virtual telescope equivalent to the distance between the individual telescopes (called the *baseline* distance). When scheduling observations on the VLTI, the time taken to reconfigure the baseline must be considered, as well as the need to run various maintenance tasks on individual telescopes making up the array.

Figure 1 The instruments on Paranal, Chile. The four larger structures are the Unit Telescopes (UTs) housing various instruments. The four smaller structures are the movable Auxiliary Telescopes (ATs). The image shows light from each AT being directed through tunnels to the central VLTI laboratory, which is housed underneath the central building. Light may also be directed from the UTs to the central VLTI equipment. Credit: ESO/L. Calçada (eso.org).

ESO uses Savile Row [8] and several of its backends to create a prototype schedule for half a year of observing time using a ranked list of science proposals as input. For each proposal, we try to allocate time on the used telescope taking into account the observability of the targets and any time constraints from the astronomer and also trying to take into account probabilistic observing conditions (e.g. weather) and deterministic conditions, e.g. moon illumination. Only proposals which can fit into the schedule will get observing time. From around 900 proposals per half-year semester, roughly a quarter are accepted and scheduled.

2 Observation Scheduling Problems

ESO has two sites in Chile hosting various optical telescopes, and jointly operates a radio telescope array at a third site. Of the optical telescope sites, the older site is La Silla (in operation since the late sixties) with the 3.6m telescope and the NTT. The newer site is Paranal with the Very Large Telescope (VLT) consisting of four 8.2m Unit Telescopes (UTs) and four 1.8m Auxiliary Telescopes (ATs). The UTs can work either independently or work together to form an interferometric array (VLTI). When the UTs operate independently, the ATs can be used as VLTI, too. A telescope has several instruments. Each instrument has specific, unique capabilities and cannot easily be moved to another telescope.

Science Proposals ESO has two Calls for Proposals per year where astronomers are invited to submit papers describing their science case and detailing the observations they wish to make. For each instrument, they list the targets they want to observe and the necessary observing conditions in *observing runs*. Observing time is allocated to individual observing runs. The proposals are reviewed for their scientific quality. Each run is assigned a *grade* and the grades are then normalised over the different instruments. The graded runs are the input for our scheduling activities. A run has the following attributes:

- A mode: Visitor Mode (VM), designated Visitor Mode (dVM), Service Mode (SM).
- Instrument and instrument setup
- Deterministic observing constraints: moon illumination ($0 \ldots 100\%$), airmass (how much atmosphere the light passes through before reaching the telescope)

- Probabilistic observing constraints: sky transparency (cloudiness: photometric, clear, thin clouds, thick clouds), turbulence category (tolerable disturbance through the air; 7 different percentages), precipitable water vapour (PWV; 0. . . 30mm)
- Optionally some time constraints, and a run grouping tag

The *observing mode* defines who makes the observations: in VM the astronomer travels to the observatory and operates the instrument herself; in dVM the remote astronomer communicates with a support astronomer at the observatory while the observations are made; and in SM, the observer prepares detailed observations in advance (*before* the observing period) and operators at the observatory carry out the observations based on current conditions. VM/dVM is *scheduled* on the calendar with a granularity of one tenth of a night, but SM is *allocated* if sufficient time under the requested observing conditions is available.

The specified airmass limits the observing window of the target. Moon illumination can be computed for each night. Sky transparency, turbulence category, and PWV have known statistical distributions. The astronomers can additionally specify time constraints such as: `between(2025-06-01, 2025-06-16)`; or `1n:0.3en (5..10) 1n:1n` (one night, followed by 0.3 nights towards the dawn, then 5 to 10 nights pause then two consecutive nights).

All runs are passed through some astronomical pre-processing that deals with all observability questions. It computes bins for the observing constraints, computes sidereal time ranges for observations, combines VM observations into compact sets of blocks for each possible start date and assigns an initial run grouping tag.

Technical Time Runs Technical Time Runs (TTRs) are used for maintenance of the telescopes or instruments (mirror re-coating, repairs, etc) or commissioning of new instruments. TTRs have a grade like science runs and may require some telescope or instruments. They can happen simultaneously if they do not need the same facilities. The time of TTRs cannot be used for science and TTRs are usually scheduled on the calendar like VM runs.

2.1 General Approach

Currently, we try to produce a schedule for each telescope plus the VLTI using different recipes involving one or more Essence Prime [7] models. The key requirement is to schedule runs according to their ranking. It is impractical to include all runs in one optimization problem: the resulting concrete model would be too large. Instead, we solve a sequence of satisfaction problems, adding runs one by one in rank order. When an instance is unsatisfiable, we discard the most recently added run and continue with the next one in rank order.

The La Silla telescopes (3P6 and NTT) are offered only in visitor mode. Time resolution is a 1/10 of a night (a “deci night”). There can be at most 2 visitors per night. Runs with the same grouping tag must be in a few sets of contiguous nights (e.g. if an instrument needs to be mounted which is a time consuming procedure; observations should happen in appropriate blocks; these blocks are an output of the model). Once all the runs that fit are known, we enable a few more constraints and try to optimise groups and runs for compactness.

On the UTs on Paranal, we have both VM and SM runs. The VM runs are scheduled similarly to the La Silla runs with one model. The SM runs are handled differently: all their observations take time from the resource pool bin of each requested observing constraint. The run is “in” only if sufficient time in the requested bin or bins with “better” conditions of *all* observing constraints is available for *all* observations in the run. This requires computing the available times in all resource bins based on which VM runs/TTRs are already scheduled (since they also use time from the bins). VM and SM runs use different models, so we need

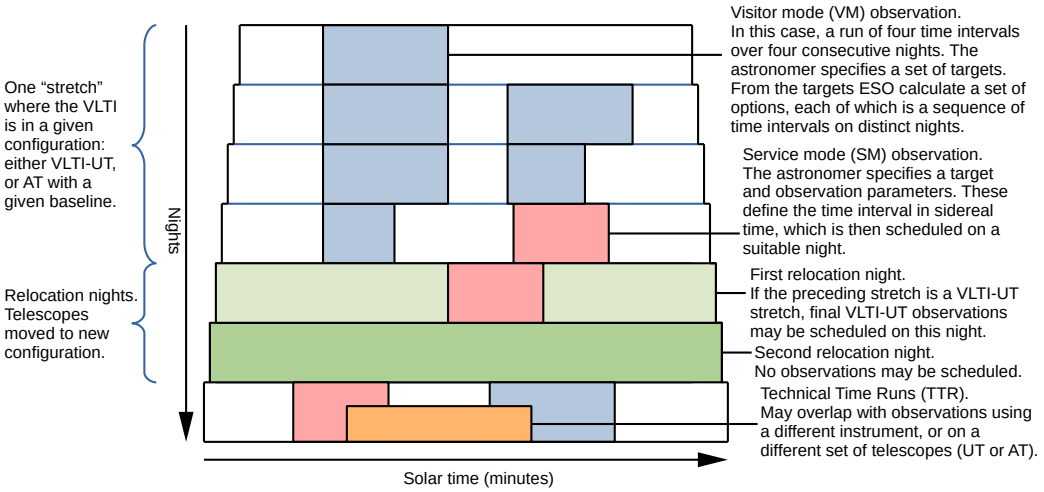

Figure 2 Illustration of VLTI scheduling showing the types of tasks and how they may overlap in time. Nights vary in length throughout the year, we have shown the nights lengthening.

to re-run the SM model (with updated resource budgets) after a new VM run is found to fit. The VM run is considered scheduled only if the SM model is still satisfiable.

2.2 Related Work

Handley et al. [4] use Integer Linear Programming (ILP) to obtain an “optimal ordering of a set of targets over a night subject to timing constraints and time-dependent slew overheads”. The problem is solved over 10 minute intervals with up to 100 targets. Naghib et al. [6] use a Markovian approach to automate schedule generation for observations lasting around 30 seconds and lens changes around 2 minutes. In a problem closer to ours, Lampoudi et al. also implement a solution based on ILP for Las Cumbres Observatory Global Telescope (LCOGT) [5] where requests have different priorities, durations, and time windows and can be fulfilled by one of a number of instruments. They report being able to select 2055 of 3864 submissions over a half-year semester. The time granularity is a hyperparameter and the exact time unit is unclear. Spohn [12] uses OR-Tools’ CP-SAT solver to schedule observations of desired targets; the *Optional Interval Variable* type is used for the no-overlap constraints.

The context for our work is described in [11], detailing the entire long-term scheduling challenge for ESO at La Silla and Paranal. Our work here specifically addresses scheduling the VLTI which presents its own unique challenges. The VLT and VLTI have a broad mix of service mode and visitor mode observations, where other observatories are mostly either all service mode or all visitor mode. In addition, there are many dependencies between observations on the VLT and VLTI that must be taken into account making the problem harder than having fewer telescopes without such dependencies.

3 VLTI Scheduling Problem

VLTI scheduling is the most difficult case (illustrated in Figure 2). Here, all observations carry a grouping tag, specifying the size of the *baseline*: small, medium, large, extended (all of these use only the ATs; VLTI-AT) or UTs (only the UTs; VLTI-UT). The baseline is the set of physical locations of the ATs. Changing the baseline takes 1 or 2 nights (depending on the two baselines) so observations are grouped accordingly and there must be sufficient time

between them. The VLTI schedule must also take all the TTRs into account that need one of the UTs, because during these times one cannot have VLTI-UT observations.

Once the VLTI has been set up with a baseline, it remains in that configuration for a set of nights (named a *stretch*). Stretches have a minimum length because it is not practical to spend one or two nights setting up a new baseline then using it for a single night. Stretches also have an upper limit designed to avoid making the UTs unavailable for long periods of time and thus blocking other observations (that are not in the VLTI schedule) on the UTs.

There are two periods called *imaging slots* (given as input) when no VLTI-UT observations occur, and which contain at least one set of nights of each of the VLTI-AT baselines. To find the baseline stretches, we attempt to put all observations somewhere on the calendar.

4 Modelling the VLTI Scheduling Problem

In this section we describe the most important aspects of each of the constraint models. Full models and all benchmark instances are provided in the experimental repository [10].

4.1 Basic Constraint Model

The decision variables are as follows:

- The night sn_i, start time ss_i (sidereal time in minutes), and end time se_i of each SM task i (where a task is part of a run).
- Equivalent variables for VM and TTR tasks: vn_i, vs_i, ve_i, tn_i, ts_i, te_i.
- The first and last nights of each TTR r: $twhen_r$ and $tlast_r$.
- The first and last nights of each VM run r: $when_r$ and $nlast_r$.
- The first and last days of each daytime TTR r: $dwhen_r$ and $dlast_r$.
- The first night of each stretch: $gbeg_{g,s}$, and the first night after the stretch ends: $gend_{g,s}$ for a group g and stretch s.

The domain of the variables representing sidereal time in minutes (e.g. vs_i) is an interval from the earliest starting time to the latest end time of a night in the schedule. For VM and TTR tasks, the value 0 is added as a dummy value to allow for optional tasks: some runs do not have a fixed number of tasks, and any excess ones are given a start and end time of 0.

Baseline Stretch Constraints Each group g represents a baseline (described in Section 3), and a stretch s of g is a set of nights when the telescopes are in the baseline of g. Stretches have a minimum and maximum length (constant for each baseline) which constrain the difference between $gend_{g,s}$ and $gbeg_{g,s}$. Within one group, any two stretches are separated by at least 4 nights to allow another stretch in between. Stretches cannot overlap, and are separated by relocation nights (illustrated in Figure 2) when ATs are moved:

$$(gend_{g_1,i} + r_{g_1,g_2} \leq gbeg_{g_2,j}) \vee (gend_{g_2,j} + r_{g_2,g_1} \leq gbeg_{g_1,i})$$

for each pair of distinct groups g_1, g_2 and stretches i, j, with r being the number of relocation nights. Within a group, stretches are constrained to be in order. Finally, a set of predefined stretches (specified by group, first night, and last night) may be given as input. These may be assigned to any *gbeg* and *gend* pair in the group as needed to satisfy the ordering constraints.

Tasks For each VM run (encompassing a set of VM tasks), a set of options is precomputed and passed in as a parameter. Each option specifies the exact night, start time and end time of all the tasks in the run. A table constraint is used to ensure the relevant task variables

(vn_i, vs_i, and ve_i for all i in the run, and $nlast_r$ for run r) take values corresponding to one of the options. VM runs are associated with a group g, and vn_i for each task in the run is constrained to be within a stretch of g (i.e. there exists a stretch that contains vn_i). TTRs are handled similarly but with a different stretch constraint: TTRs are excluded from the night before each stretch begins. Daytime TTRs are handled similarly to TTRs but only the start and end dates are needed in this case and the relevant variables are $dwhen_r$ and $dlast_r$.

SM runs are handled differently. For each run r, the parameters are the number of tasks in the run, the task duration, a time interval in sidereal time within which the tasks must be scheduled (on any suitable night), and a group. For each task i within the SM run r, the start time ss_i is constrained such that the task will start and finish within the given time interval; se_i is simply ss_i plus the given duration; and sn_i must be within any stretch of the given group (with the same constraint as for VMs).

Each task must fit within the night on which it is scheduled. The times of the start and end of each night are given as parameters *nstart* and *nend*, and for an SM task we have the constraints $nstart[sn_i] \leq ss_i \wedge se_i \leq nend[sn_i]$. The options for times of VM runs and TTRs are precalculated so their times will be within the relevant night.

Non-Overlap Constraints A key constraint is the non-overlap of tasks. For a pair of SM tasks i, j, the constraint is as follows:

$$sn_i \neq sn_j \ \vee \ se_i \leq ss_j \ \vee \ se_j \leq ss_i$$

For each pair of (SM or VM) tasks i, j, non-overlap constraints are posted if: i is an SM task and j is a VM task; i and j are SM tasks; or i and j belong to different VM runs. Within an SM run, some observations may be repeated. Tasks corresponding to repeated observations are also constrained to be in order. Within a VM run, the precalculated options for nights and times are non-overlapping so no constraint is required. Non-overlap constraints are also posted between TTRs when they are using the same telescope setup (any UT, all UTs, or all ATs) or the same instrument. UT TTRs are not allowed to overlap with any VLTI UT observations (because VLTI uses all four UTs), and the same is true for ATs. Other overlaps of TTRs with SM or VM observations are allowed.

A few instances have reserved time intervals on a specific night where no SM or VM tasks may run. For example, at the beginning of a 6-month scheduling period, some time may be reserved for installing software updates on one of the UTs. Non-overlap constraints are posted to exclude VM and SM tasks from reserved times.

Further Constraints There are a number of other constraints arising from operational policy or astronomical reasons. At most two VM observations are allowed on one night (with the exception of some VM observations that are exempted from this rule). The reason is that the control room on Paranal would become too busy with more than two visiting astronomers. The rule is implemented with an *atmost* constraint on the *vn* variables.

Another group of constraints concern imaging slots (described in Section 3; a fixed sequence of nights typically 3-5 weeks long). No VM tasks are allowed within imaging slots (a policy of the observatory); implemented as a unary constraint on each *vn* variable. For each imaging slot i, and for each AT baseline group g, there must exist a stretch of the baseline somewhere within the imaging slot; the constraint is written $\exists j : istart_i \leq gbeg_{g,j} \wedge gend_{g,j} \leq iend_i$ where $istart_i$ and $iend_i$ are the bounds of the imaging slot. Also, AT stretches must be completely outside or completely within an imaging slot, implemented as a logical combination of unary constraints on *gbeg* and *gend* variables.

There are some restrictions on the nights that activities can happen, implemented with unary constraints: the night before each stretch cannot have reserved time intervals; and VLTI UT stretches cannot be within imaging slots (because imaging slots exclusively use the ATs). Relocation nights before AT stretches must not contain TTR tasks on the ATs (implemented with $\neq$ constraints between a constant shifted *gbeg* variable and a *tn* variable).

Finally, a pair of (VM, TTR, or daytime TTR) runs a, b may have an interrun constraint that places them in a fixed order (e.g. a goes before b) and specifies a minimum and maximum separation (in nights) between the runs. Interrun constraints are implemented with $\leq$ or $<$ constraints between *vn*, *tn*, or *dwhen* variables (with some shifted by a constant).

4.2 Global Constraint Models

The VLTI problem has a set of tasks (observations, TTRs, etc) that either cannot overlap or can only overlap in restricted ways. In addition there are complex constraints on the order and timing of tasks. While we cannot capture the whole problem in a single global constraint, it is possible to capture many of the non-overlap constraints in one global constraint, reducing the size (in terms of number of constraints) and complexity of the model.

One approach would be to represent the start time of each task with a single variable (as opposed to two variables for the night and the sidereal time in minutes) and use a Disjunctive scheduling constraint [1] to capture non-overlap of VLTI observations. This would allow many of the pairwise non-overlap constraints to be removed but would create variables with very large domains (time in minutes, spanning the nights of half a year or a full year).

We decided to use a 2D packing constraint as an alternative to standard scheduling constraints. DiffN [2] prevents overlap of rectangles on a plane. Each rectangle is described by four integer variables: the x and y coordinates of one corner, and dimensions dx and dy. DiffN seems particularly appropriate here because it directly captures the fact that tasks cannot be split over two nights, and it avoids the need for variables with very large domains.

DiffN Model 1 is adapted from the basic constraint model with the following changes. Duration variables have been added for VM tasks, and connected to the night, start time and end time variables with a table constraint (recall that the possible nights and times of VM tasks are pre-calculated and provided in a table). A single DiffN constraint has been added, with the x dimension representing the night and the y dimension representing the sidereal time in minutes. All SM and VM tasks are represented as a rectangle of size $(1, dy)$ where dy is the SM or VM duration variable as appropriate. No other tasks are included in the DiffN. Finally, non-overlap constraints between VM and SM tasks are deleted, removing a quadratic number of primitive constraints.

In the first DiffN model the global constraint is quite loose because it does not include any maintenance or relocation tasks. **DiffN Model 2** is an attempt to strengthen propagation by adding more tasks to the DiffN constraint. Recall that for each relocation, the last night (or only night) of the relocation period cannot overlap with any VM or SM tasks. We use the existing variables representing the first night of each stretch (shifted by -1) to define rectangles of size $(1, dy)$ (where dy is the duration of the longest night in the schedule) and these are added to the DiffN constraint. Otherwise the model is identical to DiffN model 1.

4.3 SAT Encoding

The basic model is used when solving with SAT, and it is encoded as described in the Savile Row manual [7]. The order or direct encoding (or both) is chosen for each integer variable according to the constraints posted on the variable; constraints are encoded with the

default encodings described in the manual; and some variables (notably all se_i) are removed automatically because they are a linear mapping of another variable.

5 Experimental Evaluation

We carry out experiments to compare performance across different combinations of the models presented and backend solvers. The models are written in the Essence Prime language [7] which are read by the modelling assistant tool Savile Row [8]. Savile Row is able to tailor models to a variety of back-end solvers, including OR-Tools and SAT. We chose OR-Tools CP-SAT [9] 9.11.4210 as it has been the best all-rounder in recent constraint solving competitions [13] and it implements the DiffN global constraint. We also solve the problem with Savile Row's SAT output, using Kissat [3] 3.1.1, because in preliminary experiments this approach was usually the fastest in a single-threaded context. We provide the models, instances and software in an online repository [10].

As described in Section 2.1, each solver call is independent, solving the entire instance (i.e. scheduling a concrete set of runs, or finding that no such schedule exists). The approach may seem inefficient but it has two advantages: we do not need to represent all the proposed runs in one constraint optimisation problem (which would be very large and difficult to handle); and each instance can be optimised prior to solving by Savile Row (with a concrete set of runs, i.e. without complications such as optional tasks or reification of constraints between tasks that might hinder pre-solving). Throughout this section we use *instance* to mean one CSP instance, corresponding to one solver call and one point on the plots below (Figure 3). The instances in the repository represent the progression towards the final schedule, with instances containing incrementally more requested runs to schedule.

The experiments are run on a server with Dual AMD EPYC 7501 2.6GHz processors with 2×64 threads and 2TB RAM, running Ubuntu Linux 22.04.5. To account for randomness in the solvers, each instance is solved 5 times with different random seeds and the median total runtime recorded. We run 231 real half-year instances from ESO with these setups:

- DiffN Model 1 as described in Section 4.2, solved with OR-Tools CP-SAT single-threaded with free search;
- DiffN Model 2 as described in Section 4.2, solved with OR-Tools (as above);
- The basic constraint model, solved with OR-Tools (as above); and
- The basic constraint model encoded by Savile Row into SAT, solved with Kissat.

We also solve 110 synthetic full-year instances (including real science runs and TTRs from two successive half-years, preprocessed as a full year), using the best performing model and solver combinations for the half-year experiment, namely DiffN 1 and SAT. With the full-year instances we found that the relative performance of OR-Tools (single-threaded, free search) was poor for the most challenging instances so we also ran OR-Tools with 16 threads and without free search (because the free search flag causes it to ignore the specified number of threads). In this case we report the total CPU time across all threads.

The experimental results are plotted in Figure 3. We observe that the DiffN 1 model is slightly faster than DiffN 2 on the majority of instances, with an overall geometric mean speedup of 1.38. DiffN 1 also does marginally better than SAT, with an overall speed-up of 1.08; the advantage is more marked in the harder instances, whereas SAT is solving the easy and medium difficulty instances more quickly. The third plot shows that the global constraint model does outperform the one with decomposed non-overlap constraints, being over twice as fast on average. In terms of the full-year instances, single-threaded OR-Tools is struggling to solve the harder instances, with SAT being 1.52 times faster on average. When

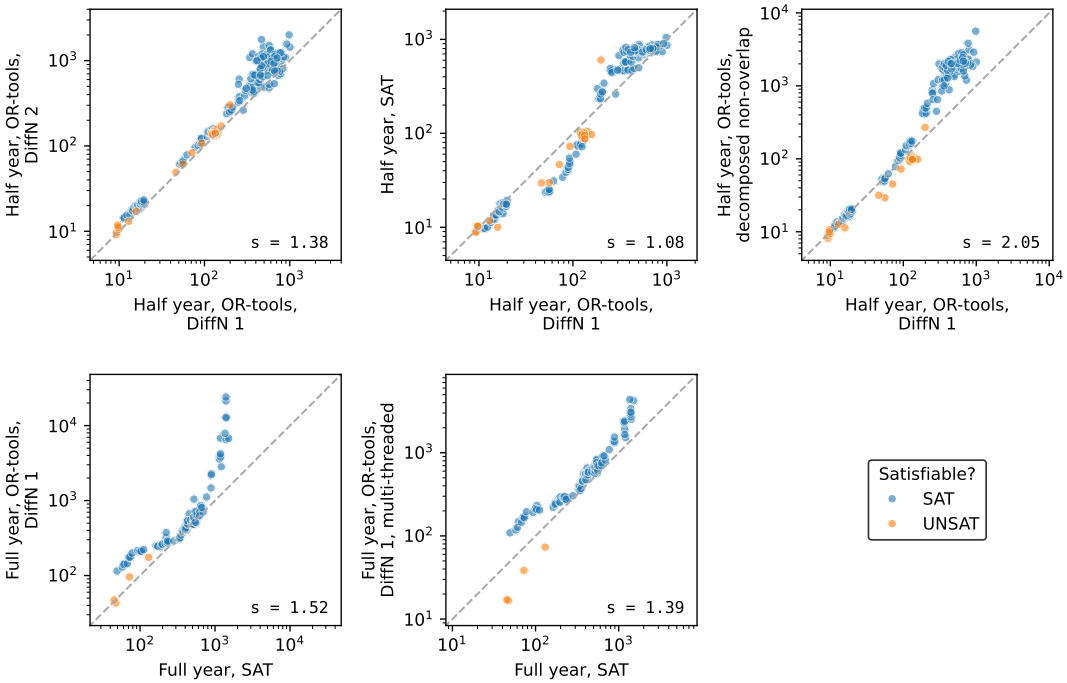

Figure 3 Experimental results. We compare total solving time (Savile Row and backend solver) in seconds for pairs of setups. Each plot displays the geometric mean speed-up s of the x-axis setup compared to the setup on the y-axis. Top: half-year schedules. Bottom: full year schedules.

running OR-Tools with 16 threads, the trend with the most challenging instances is curving away from the $x = y$ line, so it appears that SAT is scaling better overall.

There is a larger proportion of unsatisfiable instances in the half-year schedule; this is explained in part by the fact that some requested targets are barely visible during the half-year period, but there must be a set of nights when they are visible during the full year.

6 Conclusions

ESO provides state-of-the-art large telescope facilities which astronomers bid to use. They are significantly over-subscribed, leading to scheduling problems where the goal is to make the fullest possible use of the facilities while respecting priority order (based on scientific merit) on the bids. The most challenging scheduling problems arise with the VLTI facility on Paranal, Chile, where there are complex constraints between observation tasks and other tasks, and hundreds of tasks to schedule per half-year. We have presented CP models of the VLTI problem and evaluated their performance with a CP solver and a SAT solver.

The new scheduling approach replaces a laborious and partially manual process with a largely automated one, saving significant time and potentially also allowing more observations to be packed into the schedule. The new system has been in production use since the end of 2023, during which time the CP models have evolved to their current state. Future work includes refining the CP models and SAT encoding, moving to full-year scheduling, and a feasibility study of scheduling more types of observations in one model (for example, individual UT observations combined with VLTI).

References

1 Philippe Baptiste and Claude Le Pape. A theoretical and experimental comparison of constraint propagation techniques for disjunctive scheduling. In *Proceedings of IJCAI-95*, pages 600–606, 1995.

2 Nicolas Beldiceanu and Evelyne Contejean. Introducing global constraints in CHIP. *Mathematical and Computer Modelling*, 20(12):97–123, 1994.

3 Armin Biere and Mathias Fleury. Gimsatul, IsaSAT and Kissat entering the SAT Competition 2022. In *Proc. of SAT Competition 2022 – Solver and Benchmark Descriptions*, volume B-2022-1 of *Department of Computer Science Series of Publications B*, pages 10–11. University of Helsinki, 2022. URL: https://hdl.handle.net/10138/359079.

4 Luke B. Handley, Erik A. Petigura, and Velibor V. Mišić. Solving the Traveling Telescope Problem with Mixed-integer Linear Programming. *The Astronomical Journal*, 167(1):33, December 2023. doi:10.3847/1538-3881/ad0dfb.

5 Sotiria Lampoudi, Eric Saunders, and Jason Eastman. An Integer Linear Programming Solution to the Telescope Network Scheduling Problem, March 2015. arXiv:1503.07170, doi:10.48550/arXiv.1503.07170.

6 Elahesadat Naghib, Peter Yoachim, Robert J. Vanderbei, Andrew J. Connolly, and R. Lynne Jones. A Framework for Telescope Schedulers: With Applications to the Large Synoptic Survey Telescope. *The Astronomical Journal*, 157(4):151, March 2019. doi:10.3847/1538-3881/aafece.

7 Peter Nightingale. Savile Row manual, 2021. doi:10.48550/arXiv.2201.03472.

8 Peter Nightingale, Özgür Akgün, Ian P. Gent, Christopher Jefferson, Ian Miguel, and Patrick Spracklen. Automatically improving constraint models in Savile Row. *Artificial Intelligence*, 251:35–61, 2017. doi:10.1016/j.artint.2017.07.001.

9 Laurent Perron and Frédéric Didier. OR-Tools CP-SAT 9.11.4210, 2024. URL: https://developers.google.com/optimization/cp/cp_solver/.

10 Michael Prümm, Peter Nightingale, and Felix Ulrich-Oltean. Dataset for: Scheduling telescope observations for the European Southern Observatory. URL: https://doi.org/10.15124/30f68e66-acf8-44a2-ae49-54b79381c498.

11 M. Rejkuba, O. R. Hainaut, T. Bierwirth, M. Pruemm, and A. Weiss. Time Allocation and Long-Term Scheduling of ESO Telescopes at La Silla Paranal Observatory, July 2024. arXiv:2407.15470, doi:10.48550/arXiv.2407.15470.

12 Corey Spohn. *Planning Direct Imaging Observations of Exoplanets with Precursor Data*. Phd thesis, Cornell University, August 2023. doi:10.7298/hvz9-7325.

13 The MiniZinc Team. MiniZinc - Challenge 2024 Results. URL: https://www.minizinc.org/challenge/2024/results/.