Telescope Scheduling Summer Research

Lauren Dickie
Department of Engineering Science
The University of Auckland
November 2021 - February 2022

Supervisor: Oliver Sinnen

Contents

1 Introduction **4**

2 Literature Review **4**

2.1 Telescope Scheduling 4

2.2 Solution Methods 5

2.3 TSP Algorithms 6

2.4 LP/MIP Formulation 8

2.4.1 MTZ Formulation 8

2.4.2 TSP-TW 8

2.4.3 TD-TSP-TW 10

2.4.4 m-TSP-TW 11

2.4.5 TSP-mTW 11

2.4.6 Linear Ordering Formulation (Epsilon Formulation) 11

3 Problem Definition **14**

4 Problem Specifications **14**

4.1 Time-Windows 15

4.2 Slew Time 15

4.3 Expected Integration Time and Scintillation Time-Scale 16

4.4 Assumptions 16

5 Mapping to TSP **17**

6 Scheduling **18**

6.1 MIP Policy 18

6.1.1 Pre-Processing 19

6.1.2 Formulation 21

7 Implementation **23**

8 Future Work **25**

9 Appendix **29**

9.1 Scheduling Algorithms 29

9.2 Scheduling Approaches 29

1 Introduction

Telescopes have many requested observations to be made, and are a very expensive resource to build and operate. The Square Kilometer Array, the Hubble Space Telescope and the recent launch of the James Webb Space Telescope are being used to observe deeper into space and with higher sensitivity than previously possible. The development of these telescopes further drives the demand for pulsar observation and monitoring.

Creating an optimal schedule for the telescopes observation time to maximise the scientific output is very important. The development of these new telescopes further drives the need to create an optimal scheduler to observe as effectively and efficiently as possible. Literature acknowledges that there is a lack of automated dynamic schedulers for astronomical observations that can account for the complex and stochastic elements often involved [24].

This report builds upon the Part IV Project conducted by Harpreet Sing and Seong-June Her (2021) and the work conducted by Moser and Straten (2018) [25]. Both bodies of work aimed to create an algorithm for optimally scheduling pulsar observations.

Scheduling pulsar observations for a radio telescope is a stochastic and dynamic problem. The telescopes ability to be view a pulsar depends on the time, telescope location, pulsar location and the observing conditions. Observing conditions can be both predictable, such as satellite interference, and unpredictable, such as scintillation. Dynamic and stochastic conditions are very common in most astrological observations. Algorithms developed for scheduling pulsars can therefore be of use for other astrological scheduling tasks.

Moser and Straten (2018) [25] developed dynamic scheduling algorithms for observing pulsars. Further development was conducted by Singh and Her(2021) by casting telescope scheduling as a Travelling Salesman Problem (TSP). Applying TSP algorithms to telescope scheduling saw an improvement in the scheduling algorithm. This project aims to explore further TSP adaptations and solution methods to create an effective short-term dynamic pulsar scheduler.

2 Literature Review

2.1 Telescope Scheduling

The aim of telescope scheduling is the best utilise the telescope, it being a limited resource. However, quantifying what this means in mathematical terms has been very varied. Common objectives found in literature have been highlighted below, and more often than not multiple objectives were identified.

- Minimise slew time
- Maximise recording time
- Maximise the number of observations made
- Minimise the number of conflicts in the schedule
- Maximise scientific priority of completed tasks
- Fairness, Airmass, ... etc

There is a fundamental trade-off between the quality of the schedule (objective value) and its robustness (stochastic considerations) [6]. It is important to note that schedule quality can refer to many different things, depending on the objective function, and can also refer to the quality of the data obtained. It may be possible to schedule an observation at many different times, but if it is very light sensitive, some times will be preferable over others.

It is important to note that the above is generally relevant to proactive scheduling. Reactive scheduling tends to aim to minimise the disturbance to the remainder of the schedule. Often scientists are required to travel and there are dependencies between observations which is costly if disrupted [18].

It is common to see scheduling tasks broken down into short-term, mid-term and long-term. Generally different scheduling techniques are used for each time frame considered, and sometimes both an off-line and on-line scheduler are utilised for the short-term. Varying levels of interactivity is included in the scheduling model; some methods almost solely completed by a human scheduler and others fully automated. It is important to note that the specific time period each represents varies from source to source.

2.2 Solution Methods

The complexity of the constraints and the large solution space for astronomical scheduling problems, has led to many different solution approaches. Often there does not exist a feasible solution to scheduling all the requested tasks and exact solutions are too computationally expensive to generate in-real-time. Solutions have been explored for different types and location of telescope(s) as well as the type and location of the observation. For a more comprehensive analysis of existing solution methods, [24] and more recently [10] have conducted a survey of the literature to summarise the existing algorithms.

It is quite common to see telescope specific approaches, that are specialised to constraints and parameters at that site. ATIS is a command language used for optical telescope scheduling by maximising the total priority of completed tasks. The CARMENES instrument control system uses a multi-objective dynamic scheduler (CAAST). It aims to maximise scientific return and minimise telescope down time, which in the short-term simply corresponds to filtering out infeasible solutions and sorting remaining ones, first and foremost by scientific priority. ALMA (Atacama Large Millimetre/submillimetre Array) uses a online dynamic scheduling algorithm with a weighted scoring system used to sort the observations that are available to be viewed. Cherenkov Telescope Array (CTA), consisting of two cherenkov telescopes, has many operating modes and configurations of sub-arrays of telescopes. The scheulder for CTA aims to scheduled observations to maximise scientific return and minimise operating costs, using artificial intelligence for both long-term and short-term scheduling. Euclidean star is a pre-processing algorithm used to reduce the neighbourhood of each pulsar such that the solution space is much smaller [25].A more comprehensive summary of scheduling approaches can be found in Appendix 9.1

Looking at the algorithms and methods used, we can summaries the types of scheduling approaches that have been used (Appendix 9.2). If computational time allows, multi-objective scheduling problems can be solved to find an optimal solution using complete enumeration or Linear (Integer) programs. Depending on the problem specifications and constraints, other methods can also be used, such as the Critical Path Method. While it is sometimes feasible to use exact approaches for the short-term schedule, the mid-long term scheduling problem are often too

large to be solved for using exact or near exact approaches. Short-term schedules require an online approach with low computational time to adjust to the stochastic and dynamic elements of the problem. Heuristic based approaches have also been investigated for telescope scheduling, most commonly ant colony optimisation (ACO) and genetic algorithms (GA). There are also some less commonly use heuristic methods seen in literature such as Tabu Search, Evolutionary Algorithms.

2.3 TSP Algorithms

The TSP is a well studied scheduling problem with many variations to solve a range of vehicle routing, scheduling and logistics problems. To capture more complex problem constraints, many variations on the classic TSP have been developed and analysed. Variations on the objective function and the constraints can be seen in literature. As a well studied problem, it is common to see many different solution methods used for each TSP variation

TSP and variations found have been summarised in the table below, including some commonly used solution methods.

TSP Variations and Solution Methods			
Method	Problem Description	Solution Methods	Source
TSP	NP-hard problem. Minimise the travel time.	A lot of work has been done to create good heuristic algorithms to solve the TSP using methods to generate an initial solutions and ways in which it can be improved (2-opt swap, kicks).	[2]
TSP-TW	TSP but where each city can only be visited during a specified time window.	Dynamic programming approaches, Mixed integer linear programming	[5], [20]
Capacitated TSP-TW	Includes both timing windows in which cities are available for as well as capacity constraints (like that of a knapsack problem).	OR-Tools Constraint programming and routing solver.	[27]
TSP-mTW	Where there can be multiple disjoint time-windows that each node can be visited in. While this is a common problem characteristic, little literature exists.	Heuristic approach using Tabu Search, simulated annealing, variable neighbourhood search, algorithmic approach [4], [16]	
BITTP	Bi-Objective Travelling Salesman problem: To Maximise profit and minimise travelling time.	Solved by a weighted sum of the objective function using convex combinations. Convex region may not include the pareto optimal solution. Uses exploration (random search) and exploitation (2-opt swap and bit-flip) heuristics. Tours are created using the Chained-Lin–Kernighan heuristic.	[7], [31]
PC-TSP	Prize collecting Travelling Salesman Problem: Visit a subset of nodes to obtain needed prize amount while minimizing travel costs.	Lagrangian Relaxation: Provide tight bond gaps for other solution methods.	[26]
mTSP	Multiple Traveling salesman problem: Generalisation of TSP where multiple salesmen can be used in the solution (to visit the visits). Appropriate if considering a system of telescopes which could each be used to visit the locations with the addition of some constraints (like used for VRPs). Additionally to model a mid-term schedule where each salesman represents a night of viewing.	Exact methods (Integer programming, cutting planes, Lagrangian, b&b) and heuristics (adapting TSP heuristics, ANN, GA, Tabu Search etc)	[3]
TD-TSP	A single vehicle with infinite capacity i.e. find the Hamiltonian cycle while accounting for some time travel function (depending on distance but also on the position of the arc in the tour or the time period). - Generalises the Traveling Deliveryman Problem.	Branch-Price and Cut approach is the bets exact approach in literature.	[1]
OP	Orienteering problem/Selective TSP is where we aim to maximise the profits from visiting cities in a time frame. Not all the cities need to be visited but we aim to do so.	Algorithmic approximations, ACO, etc	[30]

2.4 LP/MIP Formulation

2.4.1 MTZ Formulation

The TSP problem can be described using the Miller-Tucker-Zemlin formulation. It is important to note that it is not the most effective formulation but very useful for understanding how to capture a TSP in a MIP. We start by defining x_{ij},

$$x_{ij} = \begin{cases} 1, & \text{if travel directly from node } i \text{ to node } j \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The resulting formulation uses conservation of flow and the introduction of an additional variable, u, for sub tour elimination.

$$\begin{aligned} Min: & \sum_{i=1}^{n} \sum_{j=1, j\neq i} c_{ij} x_{ij} \\ s.t. & \sum_{i=1, i\neq j}^{n} x_{ij} = 1 && j = 1, 2, ..., n \\ & \sum_{j=1, j\neq i}^{n} x_{ij} = 1 && i = 1, 2, ..., n \\ & u_i + x_{ij} \leq u_j + (n-1) * (1 - x_{ij}) && 2 \leq j \leq n \\ & x_{ij} \in \{0, 1\} && i, j = 1, 2, ..., n, i \neq j \\ & u_i \in R^+, && i = 1, 2, ..., n. \end{aligned} \tag{2}$$

We aim to minimise the travel cost. The first two constraints ensure that each node is only visited by one other node and only visits one other node. The this eliminates sub-tours i.e. if $x_{ij} = 1$ then $u_i + 1 = u_j$ where u represents the rank/order of visited nodes. The constraint shown above is the big M equivalent such that it is a linear equation.

An alternative method to eliminating the subtours is to run the MIP in a loop where if the length of the tour is not the same length as the number of nodes plus one, then add a constraint to eliminate the current solution and re-run the model.

2.4.2 TSP-TW

Formulation for TSP-TW has been presented in literature [15]. To understand how the TSP-TW is captured in The required input data is:

1. A set of nodes $\mathcal{N}$, each with an associated slew time function t: $\mathcal{N}X\mathcal{N} \rightarrow R^{>0}$ and observation time function s: $\mathcal{N} \rightarrow R^{>0}$.
2. The time windows are defined by $\mathcal{W} = \{w_o, w_1, ..., w_q, w_q + 1\}$. Where, each time-window $w \in \mathcal{W}$ is unique and defined by its start time r_w and end time s_w (there doesn't exist $w_a, w_b \in \mathcal{W}, w_a \neq w_b, r_{w_a} = r_{w_b}, s_{w_a} = s_{w_b}$).

3. A function w that assigned each node to a time-window. For a given node, $a_i \in \mathcal{N}, \mathrm{w}(a_i)$ gives the time window within which the telescopes must arrive.

Symmetry of travel times, node dependency and structured time-windows are key concepts presented in literature that help to define the problem and influence formulation.

- *Travel times* are said to be symmetric if $t(a_i, a_j) = t(a_j, a_i)$. For telescope scheduling, symmetric travel times means the slew-time from pulsar i to pulsar j is independent of which direction is travelled.

- *Node dependency*: The set E defined the connectivity between the nodes, so if every pair of nodes is connected, then $E = \{(i,j) : i,j \in [n+1]_0, i \neq j\}$. Methods of the reducing the edge set have been proposed in general scheduling settings and specifically for telescope scheduling. Work was done on this in the Part IV Project using pre-optimisation algorithms. Reference [15] present a reduction of the edge set, $\widetilde{E}$, where:

$$\widetilde{E} = \{(i,j) : \begin{cases} \text{i and j are assigned to time windows which are connected by an edge} \\ \text{OR} \\ \text{assigned to the same time window.} \end{cases} \} \tag{3}$$

  By doing so, it reduces the number of variables in the model and hence can influence the time to solve the MIP. 2

- *The time windows* for our problem aren't structured, where it is not guaranteed that for $w_a, w_b \in \mathcal{W}$, $e_{w_a} \leq s_{w_b}$ or $e_{w_b} \leq s_{w_a}$.

-

$$\begin{cases} (i,j) \in \widetilde{E}, & if (i,j) \in [n+1]_0, i \neq j, w_k = w(a_i), w_l = w(a_j), (k,l) \in D, \\ (i,j) \in \widetilde{E}, & if (i,j) \in [n], i \neq j, w(a_i) = w(a_j), \\ (i,j) \notin \widetilde{E}, & otherwise. \end{cases} \tag{4}$$

A TSP-TW formulation can be seen in equation 5 [15]. It considers weighted sum objective of minimising total travel time and minimising waiting time.

$$\begin{aligned} Min \quad & \alpha_1 \Big(\sum_{(i,j)\in E} t(a_i,a_j) x_{ij} \Big) + \alpha_2 \Big(z_{n+1} - \sum_{(i,j)\in \widetilde{E}} t(a_i,a_j) x_{ij} \Big) \\ s.t. \quad & \sum_{(i,j)\in E} x_{ij} = 1, && j \in [n+1], \\ & \sum_{(i,j)\in \widetilde{E}} x_{ij} = 1, && j \in [n]_0, \\ & r_{w(a_i)} \leq z_i \leq s_{w(a_i)} && i \in [n], \\ & z_i - z_j + (M_{ij} + t(a_i,a_j) + s(a_i)) x_{ij} \leq M_{ij}, && (i,j) \in \widetilde{E} \\ & z_i - t(a_0,a_i) x_{0i} \geq 0, && i \in [n], \\ & z_i + t(a_i,a_{n+1}) + s_i \leq z_{n+1}, && i \in [n], \\ & x_{ij} \in \{0,1\}, && ((i,j) \in \widetilde{E}, \\ & z_i \geq 0, && i \in [n+1]. \end{aligned} \tag{5}$$

It is interesting to note that the time-window constraints, also implement sub-tour elimination. Instead of z_i giving the rank of node i in the tour, it represents the time that node i is visited at.

The addition of the time-windows in the TSP links to the rising and setting time of pulsars in telescope scheduling. Most pulsars aren't available for viewing during an entire observation window and this can be captured through time-windows.

2.4.3 TD-TSP-TW

A Time dependant TSP-TW allows for the time to travel between cities to vary. Pulsars are assumed to be a constant distance apart as they travel across the sky, until one jumps out of vision and back to the start of the viewing window creating a period in which they are very far apart ([28]). The time-dependence of slew-time can be modelled using a piece-wise constant function.

[28] and [23] model a delivery routing problem using an integer linear program formulation for a TD-TSP-TW with precedence. The delivery problem aims to pick up each item from its origin node and deliver it to its destination node. The formulation presented is more complex than what is required for our problem, but give insights into how precedence, time-window, time-dependence and other constraints can be accounted for.

Formulation for TD-TSP-TW adds another dimension to x to account for time and introduce an additional variable to track the time, t.

$$x_{ij}^m = \begin{cases} 1, & \text{if travel directly from } i \in N \text{ to } j \in N \text{ and departure time is in zone } m \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

$$t_{ij}^m = \begin{cases} 1, & \text{if } x_{ij}^m = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

This is an example of the 3-index formulation where x_{ij}^k is 1 if job j is scheduled kth and immediately follows job i. Similarly, the third index here is used to indicate the time period in which the node is visited in and the jobs represent observing a pulsar.

The travel time between any two nodes, for telescope scheduling, can be assumed to follow piece-wise constant function rather than piece-wise linear, hence simplifying the formulation. Therefore, the time to travel from pulsar i to pulsar j, can be simply calculated by:

$$T_{ij}(t_i) = t_i = \sum_{j \in N} \sum_{0}^{|T_{ij}|} \sum_{m=0} t_{ij}^m \tag{8}$$

A similar but simpler formulation to the TD-TSP-TW could be used to capture much of the telescope scheduling problem. The MIP used to model the TD-TSP-TW could be adapted, formulated in Julia and solved using a solver such as Gurobi ([28], [23]). The formulation has been drawn from an unfinished paper, so verification and validation on smaller test sets would need to be conducted to ensure it enforces the desired behaviour.

2.4.4 m-TSP-TW

Another adaptation on the TSP is a multiple travelling salesperson problem with time windows. The multiple travelling salesman is analogous to having multiple telescopes to make observations in telescope scheduling. m-TSP-TW could be utilised to model observations that required multiple telescopes such as at the Submillimeter Array. However, it fails to account for the time dependence of the distance and hence slew time between the pulsars.

What is interesting to note, is that work has been done to determine upper and lower bounds for these types of problems. Bounds can be used to indicate the number of 'travelling salesmen' required to service all the nodes. This could perhaps be a useful tool in determining the complexity of the model and the constraints and adjusting hyper parameters as needed [22].

2.4.5 TSP-mTW

The TSP with multiple timing windows allows the nodes to be visited in any one of multiple-disjoint time-windows defined.

This creates constraints with an 'or' condition and can be formulated through the use of a 'switch' variable. For example, suppose there are two time windows in which t_i needs to be observed, $[r_1, s_1]$, $[r_2, s_2]$, where by definition $r_1 < s_1$ and $_2 < s_2$.

$$\begin{aligned} (r_1 - r_2)q_i + r_2 \leq t_i \leq (s_1 - s_2)q_i + s_2 \\ r_2 < r_1 \\ s_2 < s_1 \\ r_1, r_2, s_1, s_2 \geq 0 \\ q_i \in \{0, 1\} \end{aligned} \tag{9}$$

The q_i variable identifies which of the two time-windows the node is observed in. If $q_i = 1$ $r_1 \leq t_i \leq s_1$ and if $q_i = 0$, $r_2 \leq t_i \leq s_2$. The formulation will encounter issues if the inequalities between the time windows is not upheld, obtaining negative times.

It can be extended to include k time-windows where k-1 binary 'z-variables' would need to be introduced and additional constraints to ensure only one time-window is selected ie $z_{i,1} + z_{i,2} = 1$.

It is a common problem characteristic to have multiple time-windows, but one that can often be captured by adapting the time-horizon and relaxing the problem, such that not every node must be visited. Formulation allowing for multiple time windows is not very common, likely due to the increased complexity.

2.4.6 Linear Ordering Formulation (Epsilon Formulation)

All prior TSP variations had formulation based on $x_{ij} = 1$ if we traverse $arc(i, j)$ and zero if not. An alternative formulation for scheduling problems is to instead define x_{ij} as follows:

$$x_{ij} = \begin{cases} 1 \text{ if node } i \text{ is visited before node } j \\ 0 \text{ otherwise} \end{cases} \tag{10}$$

Even if some of the x_{ij} values are incorrect when solved, we can still decipher the optimal route through precedence. Therefore, the optimal solution can potentially be found faster, despite the solution containing some incorrect x_{ij} values.

There exits formulation of a TSP using this alternative definition of x, where c_{ij} is the weight of $arc(i,j)$ [17]:

$$\begin{aligned} Max & \sum_{(i,j)\in A} c_{ij}x_{ij} \\ s.t. & x_{ij} + x_{ji} = 1 && \forall i,j \in V \\ & x_{ij} + x_{jk} + x_{ki} \leq 2 && \forall i,j,k \in V \\ & x_{ij} \in \{0,1\} && \forall i,j \in V \end{aligned} \tag{11}$$

The formulation aims to maximise the weights. the constraints enforce (1) that i occurs before j or vise versa, (2) stops cycles and (3) to set the x_{ij} variable to be binary [17].

$$\begin{aligned} \text{Objective: } & Max \sum_{(i,j)\in A} c_{ij}x_{ij} - t_{n+1} \\ \text{Origin Node: } & x_{0,j} = 1 && \forall j \in V \\ \text{Destination Node: } & x_{i,n+1} = 1 && \forall i \in V \\ \text{Time: } & \text{See Below} \end{aligned} \tag{12}$$

While it is possible for us to determine the ordering, with some incorrect values, it becomes tricky to formulate the constraints with the same amount of redundancy. I had difficulty finding much literature in this area. However, I have worked to formulate the constrains provided x_{ij} is correct.

The ordering (0-n-1) can be calculated by,

$$u_j = \sum_{i=1}^{n} x_{ij} \tag{13}$$

Alternatively, we can utilise the same calculate as presented in the Miller-Tucker-Zemlin formulation, which can give the correct result, even if there are some incorrect '0' values in x_{ij}.

$$\begin{aligned} u_0 &= 0 \\ u_i + x_{ij} \leq u_j + (n-1) * (1 - x_{ij}) & \quad 2 \leq j \leq n \\ u_i \in Z_+ & \quad \forall i \in N \end{aligned} \tag{14}$$

The precedence relationship of the nodes can be given by:

$$p_j = \begin{cases} i \text{ if } u_i = u_j - 1 \\ 0 \text{ otherwise.} \end{cases} \tag{15}$$

The departure time at node j, can be calculated upon the predecessor node i (i.e. when $u_i = u_j + 1$). the slew time from i to j, the waiting time at node j and the observation time of

node j also have to be accounted for. The slew time and observation time are known and the waiting time is a variable.

$$\begin{aligned} t_0 &= 1 \\ t_j \geq t_i + T_{ij} + t_i^W + T_j^O \quad & j \in \{k \in N | u_i = u_k + 1, x_{kj} = 1\} \\ t_i^W &\geq 0 \quad \forall i \in N \end{aligned} \tag{16}$$

Based upon t_j, the time-window constraints can be enforced, as seen in prior formulations. Additionally, we can the times align with the precedence given in x_{ij} with the following. This constraint shouldn't strictly be needed. But it could be useful when developing the formulation. If the becomes binding, it could indicate an issue with the prior formulation (if slew and observation times are > 0).

$$t_i \geq t_j x_{ji} \quad \forall j \in N \tag{17}$$

In other words, if j is visited before node i, the time it is visited at must be less than or equal to the time at node i.

3 Problem Definition

Telescopes are a scarce resource with many observations being requested to be completed. Researchers submit proposals for observations, which are accepted or declined based upon the academic merit of the proposal. Once accepted, observations must be added to the schedule. Key problem characteristics for pulsar observation by a radio telescope are:

Dynamic: Input data is changing with time.
Online: Schedule updates based upon the current input data.
Stochastic: Unknown variables that can impact integration time and observation feasibility.
Single Telescopes Just a single telescope is required to make the observation.

There are two physical reasons as to why the problem is stochastic and dynamic:

1. Pulsar scintillation - can extend the integration time required or result in early abandonment due to the additional noise. The amount of scintillation is only known after a period of observing the pulsar.
2. Satellite inference - create useless observations and potentially damage the observatory equipment. Satellite locations are however predictable.

While specific to pulsars and radio astronomy, it is much the same for other astrological observations, with both predictable and un-predictable variables that impact the observations viability. We will be considering a single telescope case.

To simplify the stochastic element of the problem, a scintillation time-scale shall be used. The expected scintillation shall be used when generating the schedule. Simulation of different scenarios will account for the varying scintillation in the results.

The objective is to minimise the time that is required to visit all of the pulsars. By visiting all the pulsars in the shortest time, scientific return will be maximised.

A key assumption that has been made in prior work is that the slew-time between any two pulsars is constant. However, there are two notable cases when this doesn't hold true.

- When a pulsar is scheduled for observation before it has risen. The slew-distance is just to the pulsar's rising position, where the telescope would wait until it rises. The slew-time is shorter, but there is likely an additional wait time. Some of the pre-optimisation methods handle this by only parsing nodes have risen and so can be observed at the current time.
  When a pulsar is scheduled for observation but has already set. Then, the slew-distance is to the pulsars rising position, which could be a much larger distance. There may also be an additional wait time in this case.

4 Problem Specifications

We can consider a network graph as a representation of our problem, showing the connectivity and hence the possible paths through the network. It starts with an origin node and ends at a

destination node. The arc between two nodes, (i, j), represents travelling from node i to node j. The weight of the arc giving the time between departing node i and when node j can be departed. In the most basic case, the graph is fully connected.

Let us consider a network. $\mathcal{P}$ is the set of all pulsars to be visited $\mathcal{P} = \{p_1, p_2, ..., p_q\}$. Each pulsar can be represented by multiple nodes in the network. The mapping of nodes to pulsars, P is given in equation 18.

$$P_q = \{i \mid i \in \mathcal{N} \ \& \ \text{ is a replication of pulsar } q\} \quad \forall q = 1, 2, ..p. \tag{18}$$

The total number of nodes n in the network, used to represent the q pulsars, can thus be calculated in equation 19.

$$n = \sum_{q=1}^{p} |P_q| \tag{19}$$

Where, $|P_q|$ gives the number of elements, i.e. replications, for pulsar q.

We can define a set of n nodes, $\mathcal{N}$, where each node, $i \in \mathcal{N} \backslash \{0, n+1\}$, represents a specific pulsar and a unique time-window the pulsar is available in. Node 0 represents the origin of the current telescope position and node $n+1$ represents the dummy destination.

Each node $i \in \mathcal{N}$ has an expected integration time O_i, a time window in which observation must begin $[r_i, s_i]$ and a random scintillation timescale sc_i.

4.1 Time-Windows

We consider a fixed time-horizon, $[t_0, T]$, representing an observation period or periods. The time-window of node 0 be $[-\infty, t_0]$ and node $n+1$ be $[t_0, T]$, where t_0 is the start time.

Each pulsar p_i has a set of k time-windows, $\{(r_{i0}, s_{i0}), (r_{i0}, s_{i0}), ..., (r_{ik}, s_{ik})\}$. The pulsar can be observed during any of the time windows. Each time-window represents the a rising and setting of the given pulsar, within which observation must start. If the pulsar never sets, the pulsar can be represented by a single time window $[t_0, T]$

The time-windows for our problem aren't structured, where it is not guaranteed that for $i, j \in \mathcal{N}, r_i \leq s_j$ or $r_j \leq s_i$.

4.2 Slew Time

There is a travel time, slew time, associated with travelling between any two nodes. The slew time is non-negative and calculated based upon only the largest distance to be travelled in any one direction. It is assumed the slew speed is independent of the telescope position. Hence, the approximate slew time can be calculated based upon the distance between observations.

Additionally, it can be assumed that slew time is symmetric where the travel time from observation i to j, is the same as the travel time from observation j to i. It is reasonable to assume a

symmetric slew time because the relative movement between pulsars is negligible compared to the movement of the pulsars relative to earth.

$$t_{ij} = t_{ji} > 0 \quad \forall i, j \in \mathcal{N} \tag{20}$$

The slew time from the origin, node 0, is based upon the current telescope position and the slew-time to the destination, node $n + 1$, is zero.

4.3 Expected Integration Time and Scintillation Time-Scale

The integration time of a pulsar varied based on noise in the atmosphere. Scintillation is fluctuations in light intensity from a star as it travels through earths atmosphere due to optical turbulence. The effect of scintillation on observations is captured using a scintillation timescale sc_i, a regular time interval, after which the the scintillation is likely to have impacted the observation.

The integration time of both the origin and destination, nodes 0 and $n + 1$, is zero.

4.4 Assumptions

This research builds upon work conducted by Moser and Straten (2018) [25] and the Part IV Project conducted by Singh and Her (2021). Therefore, we assume that the calculations are accurate and the implementation is correct.

Maintenance and calibration of the telescope are not directly considered in this model. We assume that calibration causes minimal interference as it is a quick process in radio telescopes. Additionally, that maintenance can be conducted when the telescope is not in use.

It is assumed that slew-time between pulsars is irrespective of the day. If pulsar i has a slew time of t_{ij} to pulsar j today, it will have the same slew time tomorrow. Additionally, it is assumed the over-estimation of slew-time to pulsars yet to rise, has little impact on the overall schedule due to the additional wait-time.

5 Mapping to TSP

As presented in prior work (Part 4 Project Reports) we can map the telescope scheduling task to a TSP-TW problem.

Mapping to TSP-TW	
Pulsars	Cities
Telescope	Salesman
Slew Cost	Travelling Cost
Integration time	Service Cost
Rising to setting time	Time Window

This definition can be expanded to include the time dependence of the distance between pulsars. The distance between pulsars directly impacts the travel time. It is assumed that the distance is constant for a given time period and can hence be modelled by a piece-wise linear function.

Mapping to TD-TSP-mTW	
Pulsars	Cities
Telescope	Salesman
Slew Cost	Travelling Cost
Integration time	Service Cost
Rising to Setting times	multiple Time Windows
Slew Cost is a Piece-wise Constant function of time	Time Dependence

Alternatively, utilising the pulsar replications we can cast the problem as a TSP with some additional complexity. The multiple time-windows and time dependence can largely be captured by the node replications. A constraint to ensure that each pulsar can only be visited at most once and a reward for doing so in the objective need to be added.

6 Scheduling

As presented in prior work, the scheduling task can be broken down into two parts; a pre-optimisation algorithm and a dispatch policy. The pre-optimisation algorithm uses the current schedule and node to select the set of nodes, to be be neighbours and considered for the next allocation in the schedule. The dispatch policy uses this set of nodes from the pre-optimisation algorithm to decide which pulsar shall be scheduled next.

The pre-optimisation algorithms developed prior were not altered in this work. Pre-optimisation algorithms were:

1. All nodes as neighbours: All nodes in the model are considered as neighbours. Acts as a baseline pre-optimisation method.
2. Dynamic NN: Uses the Euclidean Star Algorithm to find the nearest neighbours in each direction.
3. TSP Pre-optimisation: Returns the next e earliest setting nodes.

The dispatch policies that have been explored are:

1. Random: Select one of the neighbour nodes randomly to be scheduled next.
2. Largest Slew: Select the neighbouring node with the largest slew from the current node, to be scheduled next.
3. TSP: Select the next node that creates the shortest tour from the current node.
4. TSP-TW: Select the next node that creates the shortest tour from the current node, while only scheduling nodes in their respective time windows.

The TSP-TW dispatch policy showed the most successful results across a range of data sets. To build on this success, we shall adjust the formulation through the use of the MIP Policy. By doing so, address the inaccuracies in slew-time as a result of considering just a single time-window and the assumption that slew-time between pulsars as constant.

6.1 MIP Policy

A new dispatch policy was generated by formulating the telescope scheduling problem as a MIP. Standard solvers, like those used for both the TSP and TSP-TW policies, have limited control over the formulation. Instead, we can utilise gurobi to solve the MIP and have full control over the formulation. Continuing to cast the telescope scheduling problem as a TSP, we can utilise existing TSP MIP formulations.

Instead of trying to formulate a MIP for a TS-TSP-TW, we can formulate a TSP-TW with manually duplicated cities. Each of the duplication's captures a time-window for the node and the time-dependence of the slew-time.

6.1.1 Pre-Processing

Pre-processing is required to take the input data and transform it, such that it is compatible with the MIP formulation. Additionally, it provides an opportunity to simplify the problem. The following outlines the three pre-procesisng steps taken.

1. Pulsar Replication Each pulsar has a set of associated time-windows in which it is available. In each of these time-windows, it is assumed that the distance to other pulsars is constant. Distinct nodes are formulated to represent a specific time-window for a pulsar.

Once all the required nodes have been formulated, the time windows, slew-time, and expected integration time is known.

2. Arc Reduction We can reduce the number of arcs in the underlying bi-graph though considering the feasibility to travel between nodes and defining a set of neighbours for each node that they can travel to or from. If the timing constraints do not allow travel from node i to j, based on their respective time windows, the slew time from node i to j and the observation time of node j, then we can remove the arc connecting them. Similarly, we can constrain travel for each node i by only defining arcs, (i, j), for j in a neighbourhood smaller than $j \in \mathcal{N} \backslash i$.

Both of these arc reduction techniques can be implemented by constraining x and reducing the number of variables on which we must branch. The solution space is decreased, as binary variables are forced to zero, and in turn solution time may decrease. It is important to note that adding more constraints does not always reduce solution time. Solvers, such as gurobi, are 'black boxes' making it difficult to determine if the solution time would decrease.

Methods of the reducing the edge set have been proposed in general scheduling settings and specifically for telescope scheduling. Reference [15] present a reduction of the edge set,E, where:

$$E = \{(i,j) : \begin{cases} \text{i and j are assigned to time windows which are connected by an edge} \\ \text{OR} \\ \text{assigned to the same time window.} \end{cases} \} \quad (21)$$

To implement this into the model, a simple constraint can be added as follows:

$$x_{i,j}^{k} \leq E_{i,j} \quad \forall i, j \in \mathcal{N}, \quad \forall k \in \{1, 2, .., m\} \quad (22)$$

This constraint can be used to enforce neighbourhood constraints like those which can been used prior, defining if it is possible to travel from node i to node j in time-zone k.

Currently, the pre-optimisation stage reduces the problem size, where only neighbours to the current pulsar are considered when selecting the next pulsar for the schedule. This means the model is not parsed all the information of the nodes in the network and as a result, does not fully consider impact of the current decision on future observations.

3. Time-Window Reduction Often the time-windows defined for each node can be reduced to smaller windows. Methods of reducing the time-windows, without loss of generality, can be

implemented through an iterative update of the earliest and latest arrival at each node, until no further updates are possible [11]. The following equations give the update step for each node, $i \in \mathcal{N}$:

$$\begin{aligned} r_i &= max\{r_i, min\{s_i, min_{(j,i)\in A}(r_j + T_{ji})\}\} \\ r_i &= max\{r_i, min\{s_i, min_{(i,j)\in A}(r_j - T_{ji})\}\} \\ s_i &= min\{s_i, max\{r_i, max_{(j,i)\in A}(s_i + T_{ji})\}\} \\ s_i &= min\{s_i, max\{r_i, max_{(i,j)\in A}(s_j - T_{ij})\}\} \end{aligned} \tag{23}$$

Equations represent the earliest arrival time from predecessors, earliest departure time to successors, latest arrival time from predecessors and the latest departure time to successors, respectively.

6.1.2 Formulation

We can formulate the problem as a TSP-TW with manually duplicated cities. Each of the duplication's captures a time-window for the node and the time-dependence of the slew-time.

We start by defining the flow variable, x_{ij}, where:

$$x_{ij}^{m} = \begin{cases} 1, & \text{if travel directly from } i \in \mathcal{N} \text{ to } j \in \mathcal{N} \text{ and departure time is in zone } m \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

We can relax the constraint that every node must be visited. To do so, we must introduce the variable y_i to track which nodes are visited. This relaxation is done to ensure there is a feasible solution to the problem, an issue encountered when utilising the OR-Tools solver.

$$y_i = \begin{cases} 1 & \text{If node i is visited} \\ 0 & \text{Otherwise.} \end{cases} \tag{25}$$

To enforce the time constraints of the problem, we require two additional variables: t_i and t_{ij}. t_i is non-negative and gives the time at which node i is visited, where $t_i = 0$ if node i is not visited. t_{ij}^{k} is non-Negative and gives the time that we depart node j, if travelled to from node i in time-zone k. Similarly, $t_{ij}^{k} = 0$ if we do not travel from node i to node j in time-zone k.

The formulation for TD-TSP-TW presented in literature ([28]) can be adapted to fit our scenario. We can simplify the formulation by using the replications to account for the time-dependence of slew time and multiple time-windows. Some constraints must also be added to constrain the pulsar replications. This allows us to capture additional complexity, using a simple model and consider a longer time horizon.

$$
\begin{aligned}
Min \quad & t_{n+1} - \sum_{i \in \mathcal{N}} R_i y_i \\
s.t. \quad & \sum_{j \in \mathcal{N} \setminus \{0\}} \sum_{k=1}^{m} x_{0,j}^k = 1 \\
& \sum_{i \in \mathcal{N} \setminus \{n+1\}} \sum_{k=1}^{m} x_{i,n+1}^k = 1 \\
& \sum_{i \in \mathcal{N} \setminus \{n+1\}} \sum_{k=1}^{m} x_{ij}^k = y_i && \forall j \in \mathcal{N} \setminus \{0\} \\
& \sum_{j \in \mathcal{N} \setminus \{n+1\}} \sum_{k=1}^{m} x_{il}^k - \sum_{j \in \mathcal{N} \setminus \{0\}} \sum_{k=0}^{m} x_{lj}^k = 0 && \forall l \in \mathcal{N} \setminus \{0, n+1\} \\
& \sum_{i \in P_q} y_i \leq 1 && \forall q = 1, 2, ..p \\
& t_j \geq t_{ij}^m + T_{i,j} x_{i,j}^k && \forall i \in \mathcal{N} \setminus \{2n+1\}, j \in \mathcal{N} \setminus \{0\} \\
& t_i = \sum_{j \in \mathcal{N} \setminus \{0\}} \sum_{k=1}^{m} t_{ij}^k && \forall i \in \mathcal{N} \setminus \{n+1\} \\
& w_k x_{ij}^k \leq t_{i,j}^k \leq w_{k+1} x_{ij}^k && \forall i, j \in \mathcal{N}, \quad \forall k, \forall k+1 \in \{1, 2, .., m\} \\
& r_i y_i \leq t_i \leq s_i y_i && \forall i, j \in \mathcal{N} \\
& x_{ij}^k, y_i \in \{0, 1\} && \forall i, j \in \mathcal{N}, \forall k \in \{1, 2, .., m\} \\
& t_i, t_{i,j}^k \geq 0 && \forall i, j \in \mathcal{N}, \forall k \in \{1, 2, .., m\}
\end{aligned}
\tag{26}
$$

The objective is to minimise the total tour time, with a reward for each distinct pulsar visited. Constraints 1 and 2 ensure we start at the origin and end at the destination, respectively. Constraint 3 tracks if node i is visited, using y_i and constraint 4 ensures conservation of flow. Constraint 5 ensures each pulsar is visited at most once. Constraint 6 ensures that the departure time of the successor node is greater than the predecessor plus the slew and integration time, T_{ij}. Constraint 7 gives the time calculation for each node. Constraints 8 and 9 ensure each node is visited within its assigned time zone and the time-window the node is available in.

7 Implementation

The MIP formulation was implemented in Julia, using JuMP, and in Java. In both Julia and Java, Gurobi was used to solve the model. Implementation in Julia was used for proof of concept and testing purposes. Then translated into Java, to be integrated into the existing 'telescope scheduling project'. The model was added as a new shceudler, according to the instructions given in the 'read me'. It can be selected as the scheduler by updating the 'policy_class' in the config accordingly.

The telescope scheduling project was cloned from git and the associated 'read me' followed. The following list details the steps I took to get the project working on my device (windows).

- Download MinGW64 (https://sourceforge.net/projects/mingw-w64/) and add it as a path (system environment settings - Environment variables - system variables - Path - edit - new- (paste location of MinGW64 bin)
- Right click on the project 'Open Module Settings - Libraries'. Add all the specified libraries - by download and pasting into project, then copy and then add by Maven/Java using the following:

  com.google.protobuf:protobuf-java:3.17.3
  net.java.dev.jna:jna-platform:5.8.0
  net.java.dev.jna:jna:5.8.0
  org.mongodb:bson:4.3.2
  org.mongodb:mongo-java-driver:3.12.8
  org.mongodb:mongodb-driver-core:4.2.3
- Delete all .dll and .o files from norad and all .dll from novas
- Update the 'makefile' to have the correct local path to the win64 (in folders: novas, norad).
- Download Chocolatey (https://chocolatey.org/install) and download and install the make package (run the command: choco install make).
- In intell command line, go into both the novas and norad folders and type 'make'
- Error where the path is missing the r in 'results' to write the output change the 'ín the config file to a '/' as ̊is an internal command. (To avoid, if on windows use forward slash when giving reference to folders/directories)
- To run, right click on Scheduling main (in the src/util folder) and click run.
- Alter the config to change: the data, policy, batches, scintilaton timescale etc.

I encountered a few issues (on Windows) which are noted below, along with their fixes:

- Error where the path is missing the r in 'results' to write the output. Fix: On windows, change the 'ín the config file to a '/' as ̊is an internal command - always use forward slash when giving reference to folders or directories.
- Error messages relating to Or-tools win32-x64. Fix: Also downlaod 'ortools-win32-x86-64-8.2.9025.jar', add it into the project folder and as a library to the telescope scheduling project.

Gurobi has to be added to the project for the MIP polciy to work. To do so:

- Obtain a valid Gurobi licence and follow the instillation guide provided on gurobi.com.
- Add the .jar file as a library in the telescope scheduling project (found in the lib folder).
- If gurobi can't be found in in any of the local libraries (norad, novas), then add the 'lib' and 'bin' folder as global libraries.

The MIP policy contains full functionality to allow for replications of pulsars, through the use of multiple nodes in the model. Replications can be updated through obtaining the additional time-windows, and slew-times. Additionally, the pulsar array P_q, requires updating to track which nodes represent which pulsars. Note that the first and last nodes given to the model are fixed as the origin and destination nodes.

The pre-processing steps, outlined in section 6.1.1, were actually impemented in the MIP policy as follows:

- Pulsar Replication: Create two nodes to represent each pulsar - each to represent an observation day
    1. The slew time and time-window for the first node was calculated in the same manner as the TSP-TW scheduler.
    2. The second nodes are replications of the first with the same slew-times and time-window length. The time-windows of the second pulsars start when all of the first nodes have set.
    3. The slew time between any of the first nodes and any one of the second nodes was set to a default large value of 5,000 seconds.
- Arc reduction: Not directly been applied because existing pre-processing algorithms already reduce the size fo the problem substantially.
- Time-window reduction: Implemented successfully in Julia. All the time-windows at present start at zero and are much longer than the slew and integration time. So, time-window reduction was not implemented in the MIP policy. However, it may provide benefit when the time-windows have different starting points or are shorter length in length.

8 Future Work

The MIP policy has been added to the java project with some assumptions made to formulate the replications. The next steps to improve the MIP policy and evaluate its performance are detailed below.

Acurate Data for Replications Currently, many assumptions have been made to generate replications for the pulsars. Obtaining more accurate time-windows, slew-times and integration times would give a model that is much closer to the true observation possibilities.

Pre-Optimisation for the MIP Investigate creating a tailored pre-optimisation algorithm for the MIP policy and adapt how often the schedule should be run and updated (currently only adds one node to the schedule). The pre-optimisation algorithm significantly impacts both the run time of the dispatch policy and the resulting schedule. When considering multiple time windows, the latter time-window will only be utilised if there are enough pulsars being considered such that it is needed. So, the model can decide which pulsars are best to view today and which should be viewed tomorrow. A tailored pre-optimisation algorithm to the MIP policy could be used to ensure the scheduler is making these decisions. Alternatively, the MIP policy could be run as a pre-optimisation algorithm with all the nodes prior to observation beginning. Then utilise a dynamic scheduler, to update the schedule as actual integration time is realised.

Other areas that may be off interest for further research and improvement to the scheduler have been documented below.

Multiple Telescopes Many observations rely on the use of multiple telescopes at the same time to capture the required information. Observatories, such as the Submillimeter Array, can control which and how many of the individual telescopes are used for an observation. This could be cast as a multiple travelling salesman problem, where the multiple telescopes act as the multiple people visiting the cities. Requirement constraints would be needed for each of the observations, detailing which of the 'people' or telescopes it needed and ensuring they are all provided to the city or pulsar at the same time. Additionally, each telescope can only be assigned to one observation at a given time.

MIP Solution Time Look to literature for strategies to decreasing solution time when solving TSP or network style MIP's. Some of the methods that have used in other settings are:

- Utilise break and repair heuristics: Using the LP relaxation of variables and constraints to break and 2-opt swaps and rounding the binary variables to repair the solution.
- Backwards search algorithms - can achieve exact solutions but can be difficult to add constraints to.
- Meta-Heuristics: ACO, GA

Telescope Specific Properties Look more into the movement of the telescope, if this is constant or angle dependant. Telescopes can move in two directions at once, so it is only the largest angle that determines the slew time. For each call to the scheduler, ensure the slew-time is correct. Even if the distance is small, the telescope has maximum twist so may have to 'go the long way'.

Because the MIP formulation was implemented rather than parsing the data to or-tools to solve, further complexity can be added to the model. The current pre-optimisation parses the neighbours to the current node. Alternatively, we could consider the neighbours of each of those nodes and adjust the formulation to only create a flow variable for each node to its neighbours. Rather than using a constraint, if less variables are created than the problem wont 'blow up' as fast, in terms of computational time.

References

[1] Abeledo, H., Fukasawa, R., Pessoa, A., and Uchoa, E. The time dependent traveling salesman problem: polyhedra and algorithm. *Mathematical Programming Computation 5*, 1 (2013), 27–55.

[2] Applegate, D., Cook, W., and Rohe, A. Chained lin-kernighan for large traveling salesman problems. *INFORMS Journal on Computing 15*, 1 (2003), 82–92.

[3] Bektas, T. The multiple traveling salesman problem: an overview of formulations and solution procedures. *omega 34*, 3 (2006), 209–219.

[4] Belhaiza, S., Hansen, P., and Laporte, G. A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Computers & Operations Research 52* (2014), 269–281.

[5] Boland, N., Hewitt, M., Vu, D. M., and Savelsbergh, M. Solving the traveling salesman problem with time windows through dynamically generated time-expanded networks. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (2017), Springer, pp. 254–262.

[6] Buchner, J. *Dynamic scheduling and planning parallel observations on large Radio Telescope Arrays with the Square Kilometre Array in mind*. PhD thesis, Auckland University of Technology, 2011.

[7] Chagas, J. B., and Wagner, M. A weighted-sum method for solving the bi-objective traveling thief problem. *Computers & Operations Research 138* (2022), 105560.

[8] Clarke, D., and Avarias, J. Alma scheduling: It's dynamic! *Astronomical Data Analysis Software and Systems XXI 461* (2012), 177.

[9] Colomé, J., Colomer, P., Campreciós, J., Coiffard, T., de Oña, E., Pedaletti, G., Torres, D. F., and Garcia-Piquer, A. Artificial intelligence for the cta observatory scheduler. In *Observatory Operations: Strategies, Processes, and Systems V* (2014), vol. 9149, International Society for Optics and Photonics, p. 91490H.

[10] Colome, J., Colomer, P., Guàrdia, J., Ribas, I., Campreciós, J., Coiffard, T., Gesa, L., Martínez, F., and Rodler, F. Research on schedulers for astronomical observatories. In *Observatory Operations: Strategies, Processes, and Systems IV* (2012), vol. 8448, International Society for Optics and Photonics, p. 84481L.

[11] Desrochers, M., Desrosiers, J., and Solomon, M. A new optimization algorithm for the vehicle routing problem with time windows. *Operations research 40*, 2 (1992), 342–354.

[12] Garcia-Piquer, A., Guàrdia, J., Colomé, J., Ribas, I., Gesa, L., Morales, J. C., Pérez-Calpena, A., Seifert, W., Quirrenbach, A., Amado, P. J., et al. Carmenes instrument control system and operational scheduler. In *Software and Cyberinfrastructure for Astronomy III* (2014), vol. 9152, International Society for Optics and Photonics, p. 915221.

[13] Garcia-Piquer, A., Morales, J., Ribas, I., Colomé, J., Guàrdia, J., Perger, M., Caballero, J. A., Cortés-Contreras, M., Jeffers, S., Reiners, A., et al. Efficient scheduling of astronomical observations-application to the carmenes radial-velocity survey. *Astronomy & Astrophysics 604* (2017), A87.

[14] Grim, R., Jansen, M., Baan, A., Van Hemert, J., and De Wolf, H. Use of evolutionary algorithms for telescope scheduling. In *Integrated Modeling of Telescopes* (2002), vol. 4757, International Society for Optics and Photonics, pp. 51–61.

[15] Hungerländer, P., and Truden, C. Efficient and easy-to-implement mixed-integer linear programs for the traveling salesperson problem with time windows. *Transportation research procedia 30* (2018), 157–166.

[16] Hurkała, J. Time-dependent traveling salesman problem with multiple time windows. *Annals of Computer Science and Information Systems 6* (2015), 71–78.

[17] Iranmanesh, E., and Krishnamurti, R. Mixed integer program heuristic for linear ordering problem. In *ICORES* (2016), pp. 152–156.

[18] Johnston, M. D., and Adorf, H.-M. Scheduling with neural networks—the case of the hubble space telescope. *Computers & operations research 19*, 3-4 (1992), 209–240.

[19] Johnston, M. D., Tran, D., Arroyo, B., Sorensen, S., Tay, P., Carruth, B., Coffman, A., and Wallace, M. Automated scheduling for nasa's deep space network. *AI Magazine 35*, 4 (2014), 7–25.

[20] Kara, I., Koc, O. N., Altiparmak, F., and Dengiz, B. New integer linear programming formulation for the traveling salesman problem with time windows: minimizing tour duration with waiting times. *Optimization 62*, 10 (2013), 1309–1319.

[21] Minton, S., Johnston, M. D., Philips, A. B., and Laird, P. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial intelligence 58*, 1-3 (1992), 161–205.

[22] Mitrovic-Minic, S., and Krishnamurti, R. The multiple traveling salesman problem with time windows: Bounds for the minimum number of vehicles. *Simon Fraser University TR-2002-11* (2002).

[23] Montero, A., Méndez-Díaz, I., and Miranda-Bront, J. J. An integer programming approach for the time-dependent traveling salesman problem with time windows. *Computers & Operations Research 88* (2017), 280–289.

[24] Mora, M., and Solar, M. A survey on the dynamic scheduling problem in astronomical observations. In *IFIP International Conference on Artificial Intelligence in Theory and Practice* (2010), Springer, pp. 111–120.

[25] Moser, I., and van Straten, W. Dispatch approaches for scheduling radio telescope observations. *Experimental Astronomy 46*, 2 (2018), 285–307.

[26] Pantuza Jr, G., and de Souza, M. C. Formulations and a lagrangian relaxation approach for the prize collecting traveling salesman problem. *International Transactions in Operational Research 29*, 2 (2022), 729–759.

[27] Perron, L., and Furnon, V. Or-tools.

[28] Sun, P., Dabia, S., Veelenturf, L. P., and Van Woensel, T. The time-dependent pro_table pickup and delivery traveling salesman problem with time windows. *Eindhoven University of Technology* (2015).

[29] Swanson, K., Bresina, J., and Drummond, M. Robust telescope scheduling. In *Proc. In i-SATRAS* (1994), vol. 94, pp. 347–50.

[30] Vansteenwegen, P., Souffriau, W., and Van Oudheusden, D. The orienteering problem: A survey. *European Journal of Operational Research 209*, 1 (2011), 1–10.

[31] Wagner, M., Lindauer, M., Misir, M., Nallaperuma, S., and Hutter, F. A case study of algorithm selection for the traveling thief problem. *Journal of Heuristics 24*, 3 (2018), 295–320.

[32] Wall, M. B. *A genetic algorithm for resource-constrained scheduling.* PhD thesis, Massachusetts Institute of Technology, 1996.

[33] Wegner, P., Colomé, J., Hoffmann, D., Houles, J., Köppel, H., Lamanna, G., Le Flour, T., Lopatin, A., Lyard, E., Melkumyan, D., et al. Simultaneous operation and control of about 100 telescopes for the cherenkov telescope array. In *Journal of Physics: Conference Series* (2012), vol. 396, IOP Publishing, p. 012052.

9 Appendix

9.1 Scheduling Algorithms

9.2 Scheduling Approaches

Scheduling Algorithms		
Method	Description	Source
ATIS	Popular command language used to describe the execution/scheduling on automated optical telescopes. Maximises the sum of priority of completed tasks.	[6]
Associate Principle Astronomer(APA)	No human Interaction. Scheduler updates the whole schedule in the event of a change. Feasible tasks are selected based on a set of criteria (priority, number of observations remaining, etc)	[6]
CEERES	'Temporal-look-ahead'. Implemented alongside ATIS. Aims to maximise a weighted linear function by exploring alternative solutions and storing the best so far. Time consuming.	[6]
Just-in-Case (JIC)	'Multiply Contingent Schedule'. Rather than a dynamic schedule, predict where the schedule is likely to break and generate a new schedule for that case.	[29], [6]
genH	Mutates an existing schedule, finds the best mutation and schedules it with APA to test its score. Keeps the overall bets schedule.	[6]
NightWatch	A tactical planning tool for the royal Greenwich observatory and shot term scheduler for optical astronomy. Based on a CSP resolver with delayed evaluation and forward checking.	Spragg and Smith (1993)
GMRT	Linear solver to give a first cut (potentially infeasible) solution. Repaired using heuristics and human interaction	Charote et al. 2009.
Spike	Software for scheduling the Hubble Space Telescope. Uses a Repair Heuristic for large scale CSP problem through value ordering to minimise the number of constraint violations.	Miler et al 1988, Minton et al 1992.
ROAST	Relax the problem to a generalised assignment problem and then tasks can be ordered based upon priority.	
NASA	Use a depth first search and relaxations to generate a schedule. Remaining conflicts are resolved through collaboration.	[19]
Euclidean Star	Consider just the nearest neighbours in each direction i.e. determine if it is more efficient to travel to a pulsar directly or via another pulsar.	[25]
CARMENES (CAST)	A dynamic scheduler using Global and Local search approach (Genetic Algorithms and Hill Climbing) with an online and off-line scheduler. Objective is to minimise time not utilised and maximise scientific output. Mid-Long Term: GA. Short-term: Simple filter out infeasible observations then sort.	[12] [13]
CTA (Cherenkov Telescope Array)	A scheduler for the two Cherenkov telesocpes; one in each hemisphere. Uses artificial intelligence (Guarded Discrete Stochastic Neural Network) to maximise scientific return and minimise operating cost for both long and short term scheudles.	[33], [9]
ALMA (Atacama Large Millimetre/submillimetre Array)	Weighted scoring system to dynamically sort available tasks.	[8]

Summary of Scheduling Approaches		
Method	Description	Source
Complete Enumeration	Guaranteed optimality if deterministic. Requires high computational cost - Not computationally feasible for our problem.	
JobShop	Can be solved to or near optimally using developed algorithms but only considering a single objective.	
LP/IP/MIP	Guaranteed optimality if deterministic. Requires high computational cost. Methods such as column generation and constraint satisfaction problem relax the problem and reduce computational time. Has been used successfully in problems such as Airline Scheduling. Can be cast as a CSP by only considering the feasibility constraints and discretising time into intervals. Repair heuristics and CSP for large scale task scheduling have also shown effective.	[18], [21]
TSP	Project the problem into a well-studied domain. Many algorithms and methods have been developed. Focus of the P4P and showed promisisng results.	
ACO	Commonly used for shortest path problems i.e. TSP. Easily control exploration and exploitation. Provides a good starting point but hard to adapt the algorithm for complex constraints.	
GA	Using evolution to adapt a known solution to produce a potentially better one. Memetic Genetic algorithms seem to bee preferred for task scheduling due to the implementation to avoid early convergence [10].	[32], [14], [13]
SA	Seems to be an approach that has been tried but relatively unsuccessfully in terms of algorithm performance (high time to solve). Used for job-shop scheduling problems.	[3]
Probabilistic	Used to account for stochastic variables in the problem to select the schedule that is expected to yield the highest objective value.	
Knapsack	Account for the limited resource by modelling it as a knapsack problem	[6]
ANN	Neural Networks have been investigated. Difficult problem to project into a ANN input/output space. Simpler heuristics have been shown to outperform ANN. Requires the continuous time to be discretised.	[18]