Scheduling High-Cadence Telescope Observations

An optimization approach

João Pedro Pedroso

INESCTEC and Faculty of Sciences, University of Porto

Joint work with:

- Shiro Ikeda, ISM
- Tomoki Morokuma, The University of Tokyo
- Shigeyuki Sako, The University of Tokyo

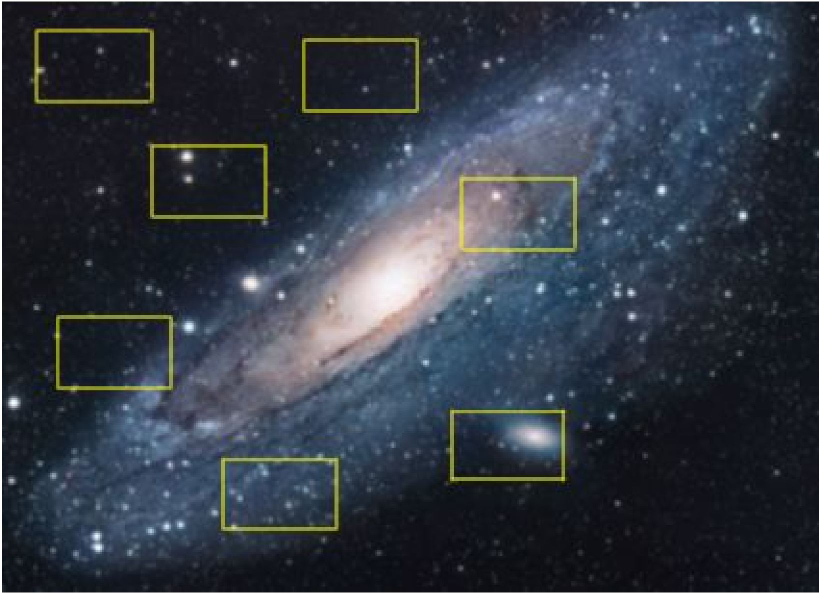
Kiso Symposium, Kiso, July 2019

The problem

Situation (my understanding):

- Telescope used for detecting supernovae right after explosions
  - rapid increase in observed flux, requiring multiple observations during a night
- Strategy:
  - take successive images of a given zone
  - check for differences between them
- In this context:
  - try to observe the whole visible celestial sphere
  - repeat some time later
  - there must be a minimum delay between successive images
  - aim: maximize the number of observations made
  - in other words, minimize the time lost
    - telescope movements
    - waiting time

Background: optimization tools



Background: optimization tools

Consider the following situation:

- 7 positions to observe in the sky
- Each position
  - has an expected reward
  - requires a certain time to be photographed
- A telescope is available for a limited time

Example

- Data:

Position:	1	2	3	4	5	6	7
Reward:	7	2	4	9	1	2	3
Time:	12	8	11	19	5	2	5

- Total available time: 30
- How can we solve the problem?

Example

- Data:

Position:	1	2	3	4	5	6	7
Reward:	7	2	4	9	1	2	3
Time:	12	8	11	19	5	2	5

- Total available time: 30
- How can we solve the problem?
- Mathematical formulation: *knapsack problem*
  - Variables: $x_1, x_2, x_3, x_4, x_5, x_6, x_7$
    - $x_i = 1$ if we photograph position i, 0 otherwise
    - binary variables, constrained to values 0 or 1
  - Objective:
    - maximize $7x_1 + 2x_2 + 4x_3 + 9x_4 + x_5 + 2x_6 + 3x_7$
  - Constraint:
    - subject to $12x_1 + 8x_2 + 11x_3 + 19x_4 + 5x_5 + 2x_6 + 5x_7 \leq 30$

Mathematical formulation

Knapsack problem: more concisely:

$$\begin{aligned} \text{maximize } & \sum_j v_j x_j \\ \text{subject to } & \sum_j w_j x_j \le W \\ & x_j \in \{0, 1\} \qquad \forall j \end{aligned}$$

How to solve it – with Gurobi and Python

Simply describe the problem, and send it to a general purpose solver

```
from gurobipy import *
m = Model()
x = {}
for i in range(1,8):
    x[i] = m.addVar(vtype="B")
m.addConstr(12*x[1] + 8*x[2] + 11*x[3] + 19*x[4] + 5*x[5] + 2*x[6] + 5*x[7] <=
m.setObjective(7*x[1] + 2*x[2] + 4*x[3] + 9*x[4] + x[5] + 2*x[6] + 3*x[7], GRB.
m.optimize()
for i in range(1,8):
    print(x[i].X)
```

How to solve it – with Gurobi and Python

Simply describe the problem, and send it to a general purpose solver

```
from gurobipy import *
m = Model()
x = {}
for i in range(1,8):
    x[i] = m.addVar(vtype="B")
m.addConstr(12*x[1] + 8*x[2] + 11*x[3] + 19*x[4] + 5*x[5] + 2*x[6] + 5*x[7] <=
m.setObjective(7*x[1] + 2*x[2] + 4*x[3] + 9*x[4] + x[5] + 2*x[6] + 3*x[7], GRB.
m.optimize()
for i in range(1,8):
    print(x[i].X)
```

```
Optimal solution found (tolerance 1.00e-04)
Best objective 1.600000000000e+01, best bound 1.600000000000e+01, gap 0.0000%
1.0
0.0
1.0
0.0
0.0
1.0
1.0
```

How to solve it – with a modeling language

Simply describe the problem in a modeling language, and send it to a general purpose solver

```
ampl: var x {1..7} binary;
ampl: maximize z: 7*x[1] + 2*x[2] + 4*x[3] + 9*x[4] + x[5] + 2*x[6] + 3*x[7];
ampl: subject to Capacity:
      12*x[1] + 8*x[2] + 11*x[3] + 19*x[4] + 5*x[5] + 2*x[6] + 5*x[7] <= 30;
ampl: solve;
```

How to solve it – with a modeling language

Simply describe the problem in a modeling language, and send it to a general purpose solver

```
ampl: var x {1..7} binary;
ampl: maximize z: 7*x[1] + 2*x[2] + 4*x[3] + 9*x[4] + x[5] + 2*x[6] + 3*x[7];
ampl: subject to Capacity:
      12*x[1] + 8*x[2] + 11*x[3] + 19*x[4] + 5*x[5] + 2*x[6] + 5*x[7] <= 30;
ampl: solve;
```

```
Academic license - for non-commercial use only
Gurobi 8.0.1: optimal solution; objective 16
2 simplex iterations
1 branch-and-cut nodes
ampl: display x;
x [*] :=
1  1
2  0
3  1
4  0
5  0
6  1
7  1
;
```

How to solve it?

General-purpose optimization solvers:

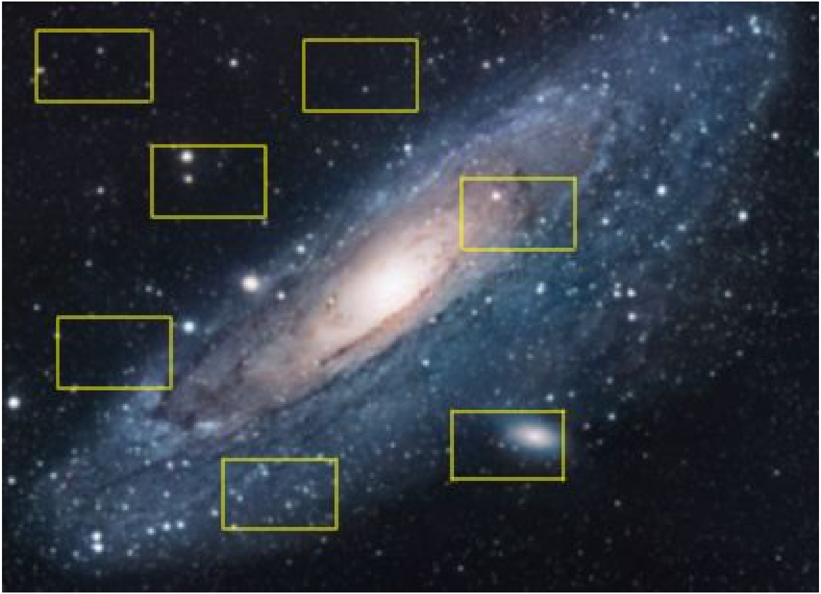
- No need to know what methods are used for solving
- Very powerful:
  - most of the underlying optimizatin problems are NP-hard
    - in the worst case, take exponential time in terms of the size of the problem
  - but in practice, even very large problems can be solved
    - often, thousands or millions of variables and/or constraints
- Convenient way to get a proven optimum
  - even open source solvers involve years of development

The problem

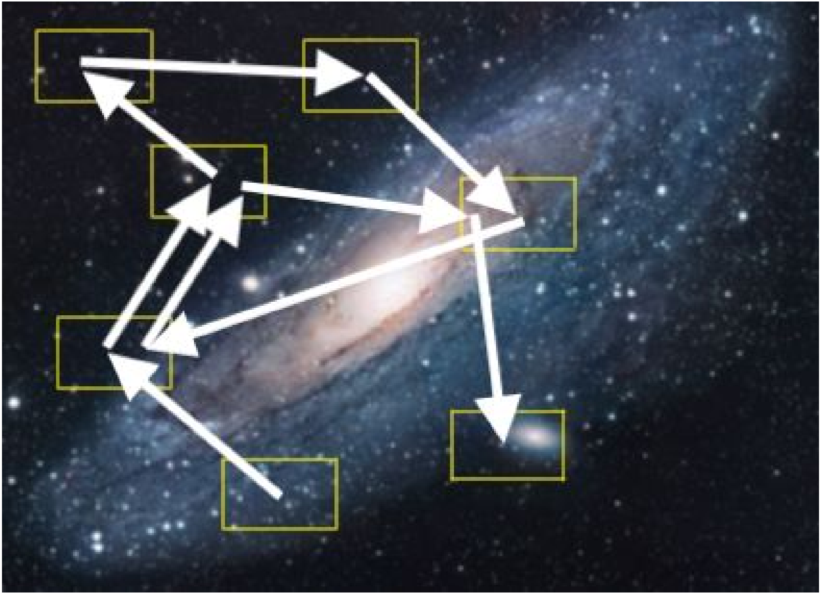
The problem

- There is a set of positions to be observed in the sky
- Each of them can be observed on a given configuration of the telescope
- We want to
  - minimize unproductive time
  - maximize the number of positions observed 3 times during the night
- Difficulty: sky "moves" during the night
  - setup between two telescope positions is time-dependent

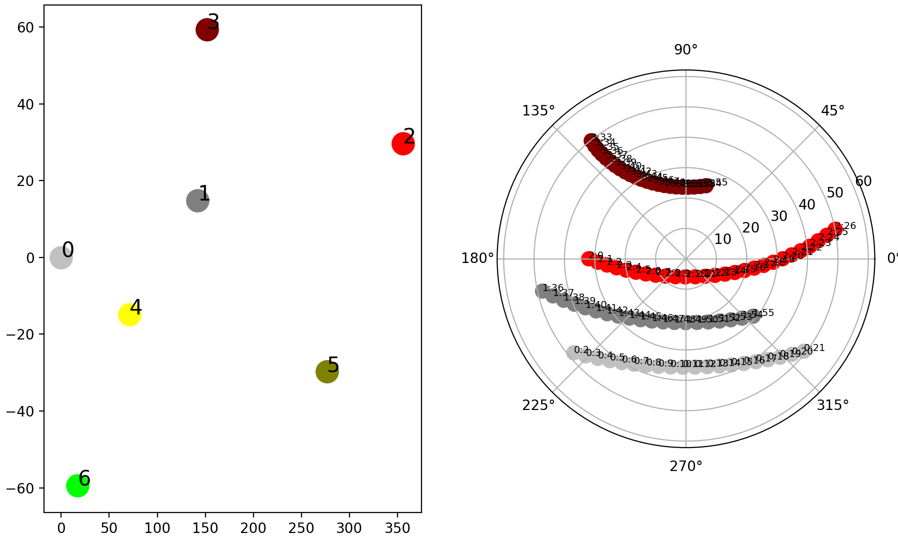
Background



Background



Figure



An optimization model

An optimization model

$$
\begin{aligned}
\text{maximize} \quad & \sum_{k \in K} z_k \\
\text{subject to} \quad & \sum_{i \in I} x_{it} \le 1 && \text{for } t = 0, \ldots, T \\
& x_{i,t-1} = \sum_{j \in I} w_{ijt} && \forall i \in I, t = 1, \ldots, T \\
& x_{jt} = \sum_{i \in I : t - c_{ij} > 0} w_{ij,t-c_{ij}} && \forall j \in I, t = 1, \ldots, T \\
& y_{k0} = 0 && \forall k \in K \\
& y_{kt} \le \sum_{i \in I} a_{ikt} x_{it} && \forall k \in K, t = 1, \ldots, T \\
& \sum_{t'=t}^{\min(T, t+d_k)} y_{kt'} \ge d_k (y_{kt} - y_{k,t-1}) && \forall k \in K, t = 1, \ldots, T \\
& z_k \le \sum_{t=1}^{T} y_{kt} && \forall k \in K
\end{aligned}
$$

(all variables are binary)

Data

- $K \rightarrow$ set of positions to be observed in the sky
- $I \rightarrow$ set of positions in the telescope
- $T \rightarrow$ number or periods to consider (time discretization)
- $a_{ikt} \rightarrow$ connect telescope and sky's positions:
  - $a_{ikt} = 1$ if at period t telescope in position $i \in I$ observes sky's position $k \in K$
  - $a_{ikt} = 0$ otherwise
- $c_{ij} \rightarrow$ time necessary to move the telescope from position i to j
- $d_k \rightarrow$ time necessary to make observation at sky's position k

Variables

- Main decision variables:
  - $x_{it} = 1$ if telescope is on position i at period t
  - $x_{it} = 0$ otherwise
- Telescope movement:
  - $w_{ijt} = 1$ if at period t telescope moves from position i to position j (possibly, $j = i$)
- Observed: (determined in terms of x)
  - $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- Positions observed: (determined in terms of y)
  - $z_k = 1$ if sky's position k has been observed

Constraints (#1)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

At each period, telescope is (at most) in one position

$$\sum_{i \in I} x_{it} \leq 1 \qquad \text{for } t = 0, \ldots, T$$

Constraints (#2)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

If the telescope was in position i at $t-1$, then at t it must move to some (possibly the same) position

$$x_{i,t-1} = \sum_{j \in I} w_{ijt} \qquad \forall i \in I, t = 1, \ldots, T$$

- if $x_{i,t-1} = 1$, then one of the w_{ijt} must be non-zero

Constraints (#3)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

For being in position j at period t, the telescope must have been in a position i (possibly the same) early enough to move to j

$$x_{jt} = \sum_{i \in I : t - c_{ij} > 0} w_{ij,t-c_{ij}} \qquad \forall j \in I, t = 1, \ldots, T$$

Constraints (#4)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

No observations can be made at $t = 0$

$$y_{k0} = 0 \qquad \forall k \in K$$

Constraints (#5)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed
- $a_{ikt} \rightarrow 1$ if at period t telescope in position $i \in I$ observes sky's position $k \in K$

Observing sky's position k at period t is only possible if the telescope is in a position from which k can be observed

$$y_{kt} \leq \sum_{i \in I} a_{ikt} x_{it} \qquad \forall k \in K, t = 1, \ldots, T$$

Constraints (#6)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed
- $d_k \rightarrow$ time necessary to make observation at sky's position k

If an observation at point k has started in period t, then the same position must be observed at least d_k successive periods

$$\sum_{t'=t}^{\min(T,t+d_k)} y_{kt'} \geq d_k(y_{kt} - y_{k,t-1}) \qquad \forall k \in K, t = 1, \ldots, T$$

- observing point k starts in period t iff $y_{k,t-1} = 0$ and $y_{kt} = 1$
- in that case, the right-hand side is positive
- otherwise, the constraint becomes redundant

Constraints (#7)

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

A position is counted in the objective only if it was observed at some valid period

$$z_k \leq \sum_{t=1}^{T} y_{kt} \qquad \forall k \in K$$

Objective

- $x_{it} = 1$ if telescope is on position i at period t
- $w_{ijt} = 1$ if at period t telescope moves from position i to position j
- $y_{kt} = 1$ if sky's position k is observed at period t, 0 otherwise
- $z_k = 1$ if sky's position k has been observed

Objective: maximize the number of positions observed:

$$\text{maximize} \sum_{k \in K} z_k$$

Refinements: second-time observations

- What happens if all the positions can be observed?
- We should take into account second-time observations
  - also third-time, fourth-time, ...
- Additional variables:
  - $y'_{kt} = 1$ if position k is observed for the second time at some period t
  - $y'_{kt} = 0$ otherwise

Refinements: second-time observations

- A minimum number of periods (Δ) must elapse since the first observation
- In other words: y'_{ks} *must be zero for* Δ *periods after period* t *at which* y_{kt} *changed from 1 to 0*
- Additional constraints ($\forall k \in K, t = 1, \ldots, T$):

$$
\begin{aligned}
y'_{kt} &\leq 1 - (y_{k,t-1} - y_{kt}) \\
y'_{k,t+1} &\leq 1 - (y_{k,t-1} - y_{kt}) \\
&\ldots \\
y'_{k,t+\Delta} &\leq 1 - (y_{k,t-1} - y_{kt})
\end{aligned}
$$

- A new variable z'_k is needed for counting the number of second-time observations (as with z_k)
- Extension for three-times observations: z''_k

Objective: maximize the number of three-times observations

$$\text{maximize} \sum_{k \in K} z''_k$$

Issues

- The previous model is good, but. . .
- Is it acceptable in practice?

Issues

- The previous model is good, but...
- Is it acceptable in practice?
- For a typical instance:
  - sky positions: $> 300 \rightarrow \sim 100000$ arc variables
  - time discretization:
    - each image: ~ 48 seconds
    - each movement: from a few seconds to ~ 1 minute
- If we discretize to 1 second: > 4000 million variables...

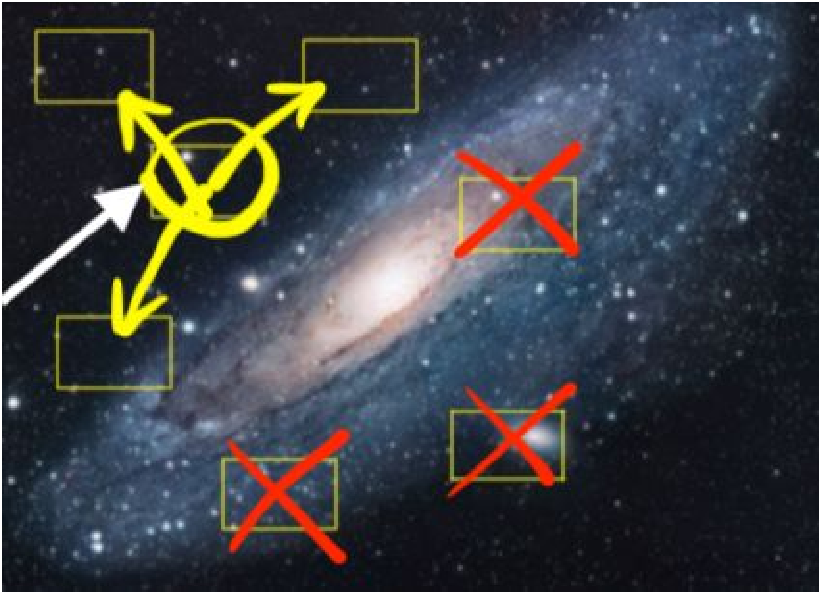
Practical approach # 1

Practical approach # 1

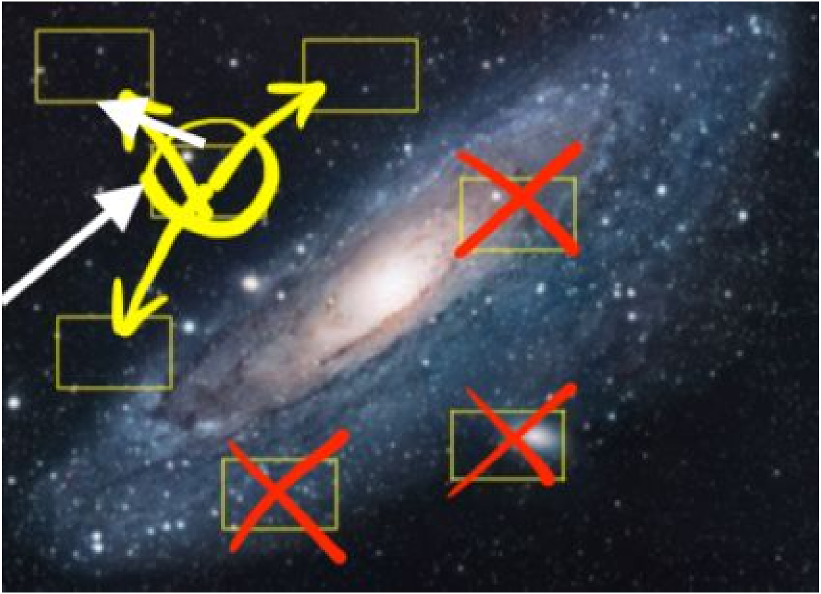
For dealing with the practical problem:

- Motivation: as we cannot afford much detail on future data, concentrate on the next movement
- Very simple idea: use a *nearest-neighbor approach*
- Well known heuristic method for the traveling salesman problem (TSP)

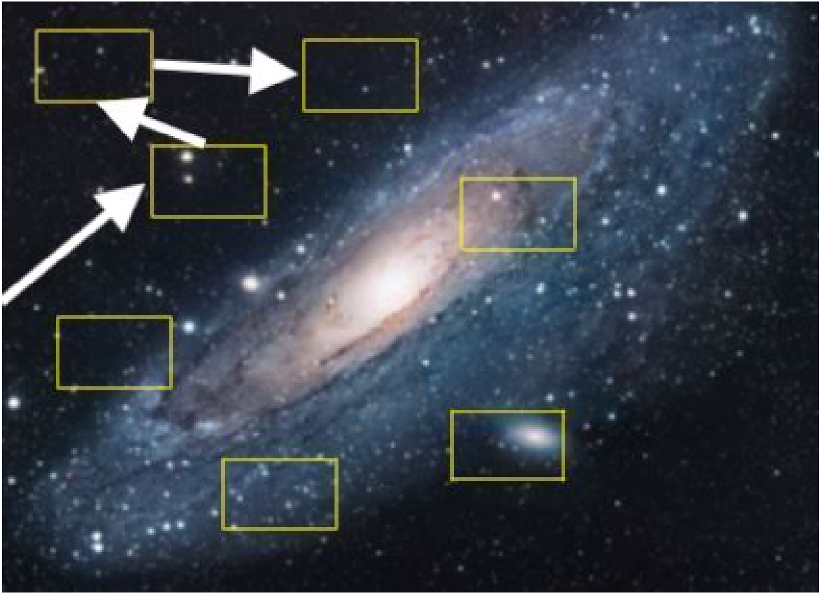
Nearest-neighbor



Nearest-neighbor

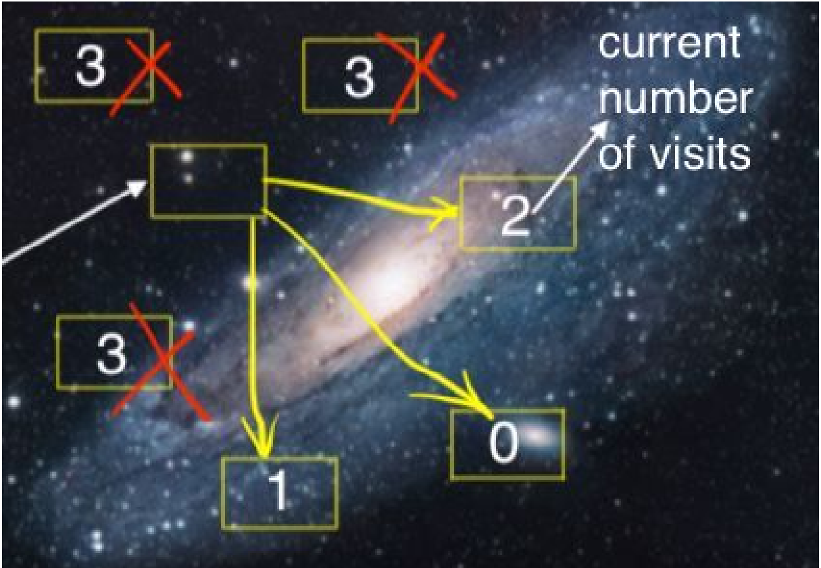


Nearest-neighbor



Nearest-neighbor: improvement

- Consider only neighbors visited at most $N + 2$ times, where N is the minimum number of visits



Algorithm: nearest-neighbor

Solution contruction procedure:

- select (arbitrarily) a visible point
- repeat:
  - move to closest "visitable" point
    - visible and with minimum delay from previous observation
  - advance simulation time: *movement* + *exposure durations*
  - update set of "visitable" points
  - determine distance from current point to all visitable

Algorithm: nearest-neighbor

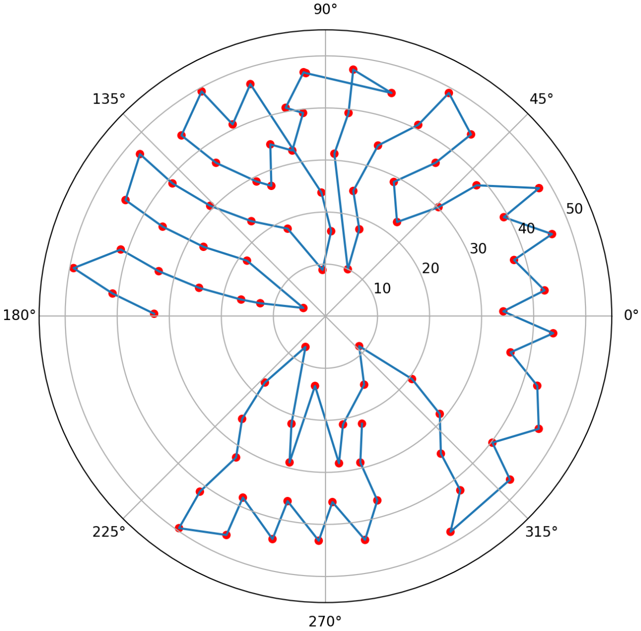
Solution contruction procedure:

- select (arbitrarily) a visible point
- repeat:
  - move to closest "visitable" point
    - visible and with minimum delay from previous observation
  - advance simulation time: *movement + exposure durations*
  - update set of "visitable" points
  - determine distance from current point to all visitable

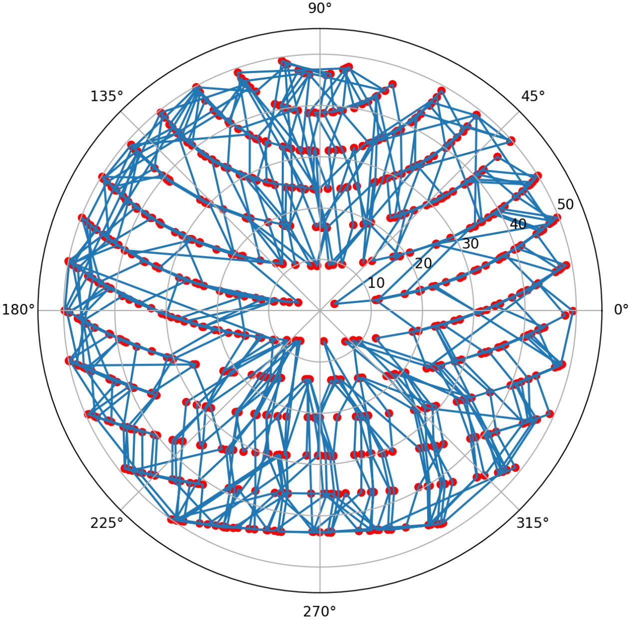
These solution constructions can be iterated:

- choose all different starting points
- for each of them, construct a solution starting from thene
- generates many solutions
- at the end, choose the best of them

Initial part of the solution



Full solution

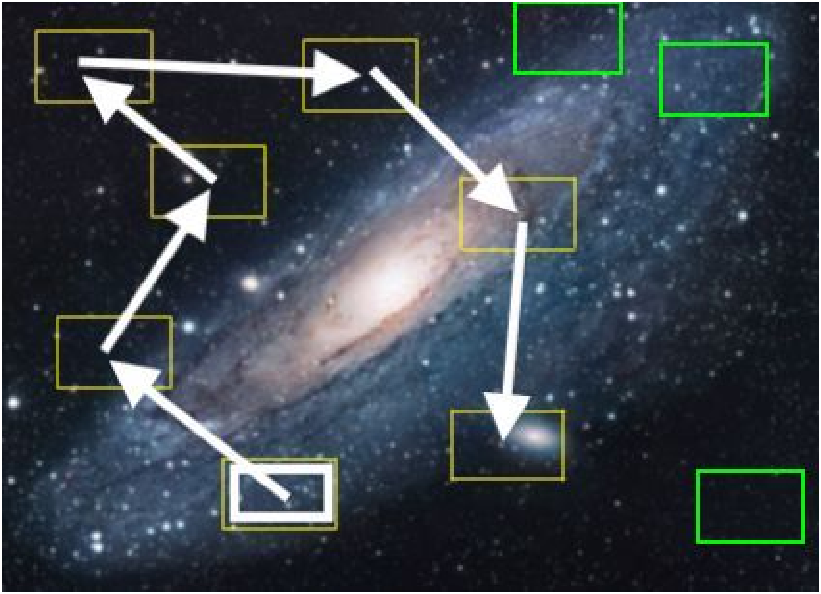


Practical approach # 2

Practical approach # 2

- Nearest-neighbor is blind
  - considers only the next step
- Can we improve it?
  - rolling-horizon

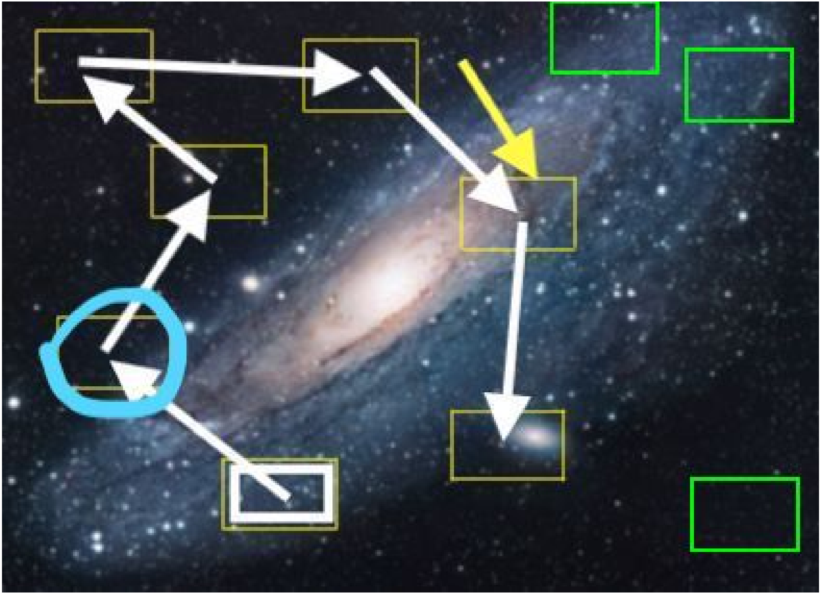
Practical approach # 2



Rolling-horizon

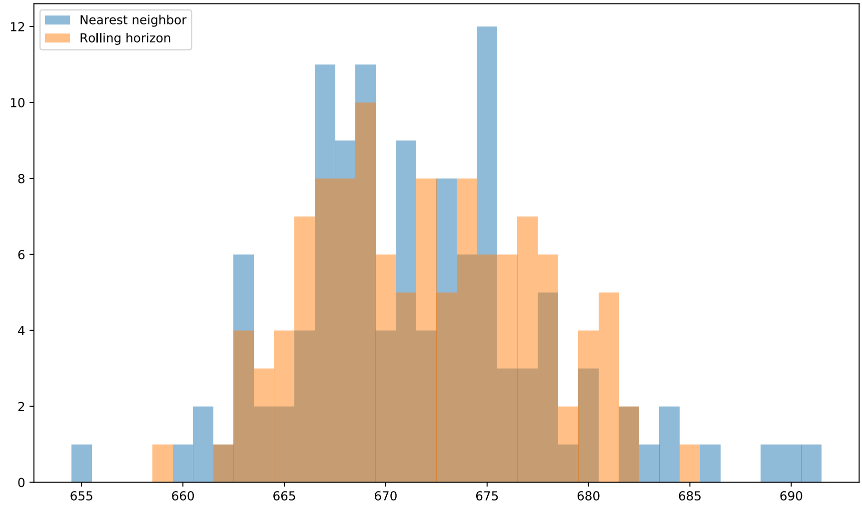
- Consider current position of the telescope
- Determine the N closest observable point
- Schedule them optimally
  - approximate dynamics of the movement between two celestial positions
  - consider present movement times
  - use optimization model for the TSP
- Commit only to the next point to visit

Rolling-horizon

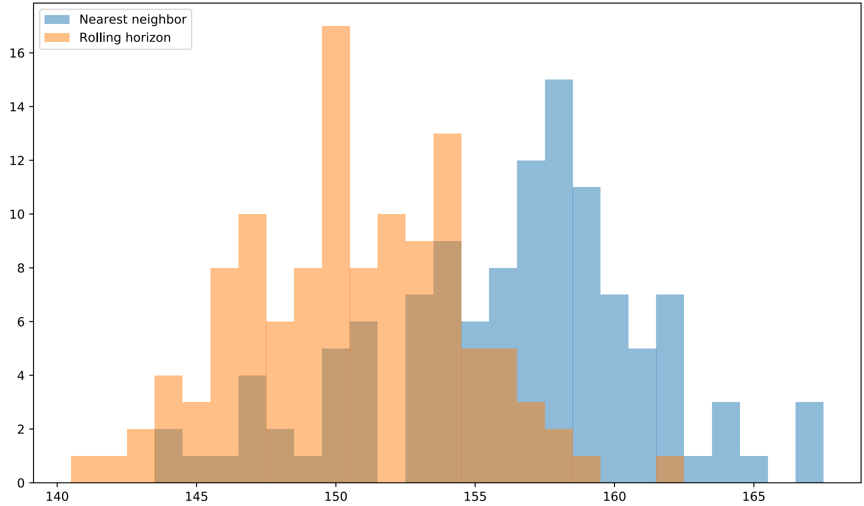


Analysis

Histogram for the total number of observations



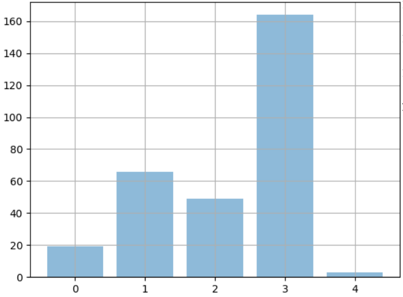
Histogram for the number of 3-times observations

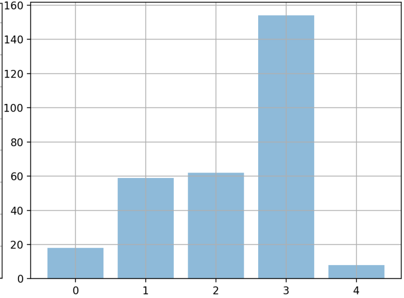


Histogram: # n-th observations (best solution)

nearest neighbor

rolling horizon





Comparison:

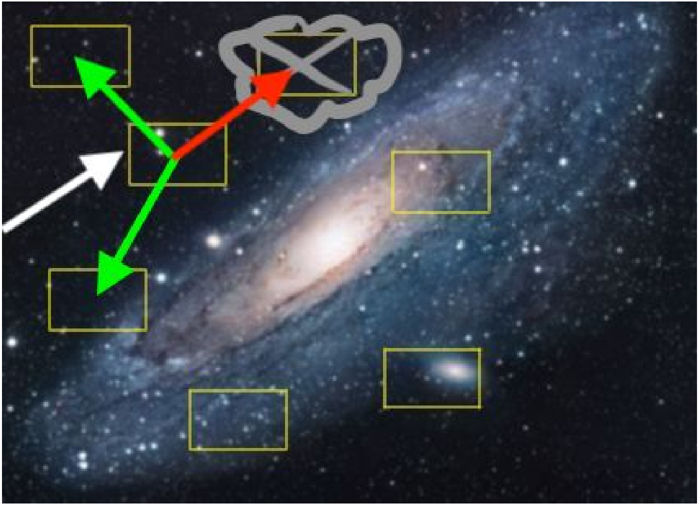
- Rolling-horizon heuristic:
  - makes a better usage of the time
  - allows more observations overall
- Nearest-neighbor heuristic:
  - less consistent
  - greater variability on solutions constructed
  - allows more 3-times observations
- If distance independent of observation time:
  - nearest-neighbor constructs hundreds of solutions in just a few seconds
  - good for reacting in real-time

Further issues

- Real time data:
  - weather conditions: clouds may obstruct observation
  - use whole sky image analysis to select observable points

Further issues

- Real time data:
  - weather conditions: clouds may obstruct observation
  - use whole sky image analysis to select observable points

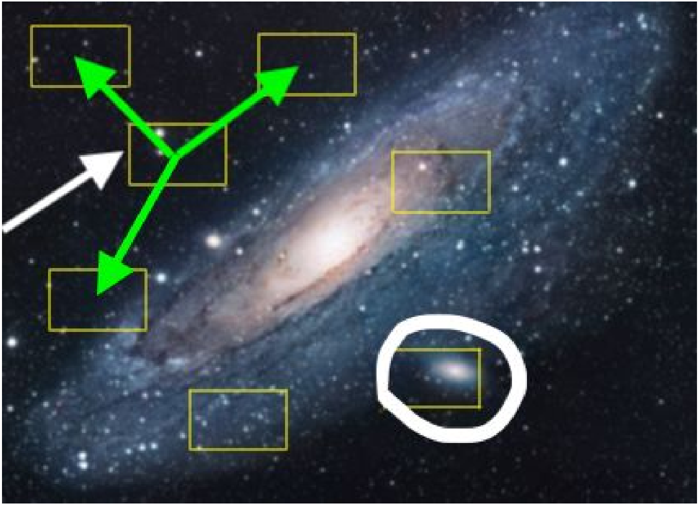


Further issues

- Force some observations, *e.g.*
  - observe area around gravitational wave
  - follow an asteroid

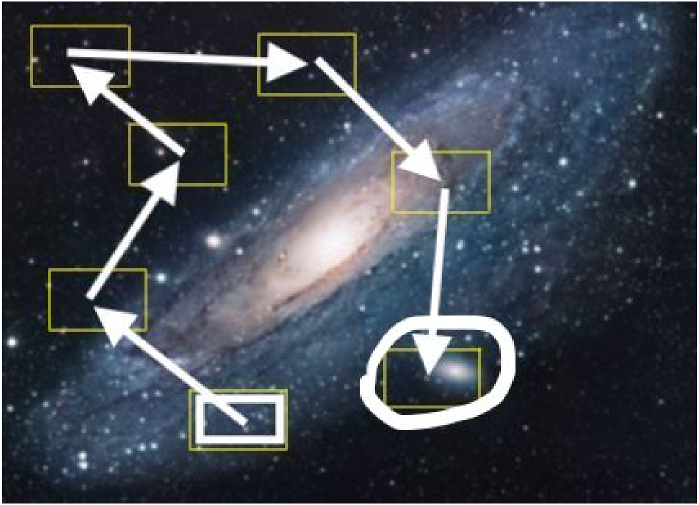
Further issues

- Force some observations, *e.g.*
  - observe area around gravitational wave
  - follow an asteroid



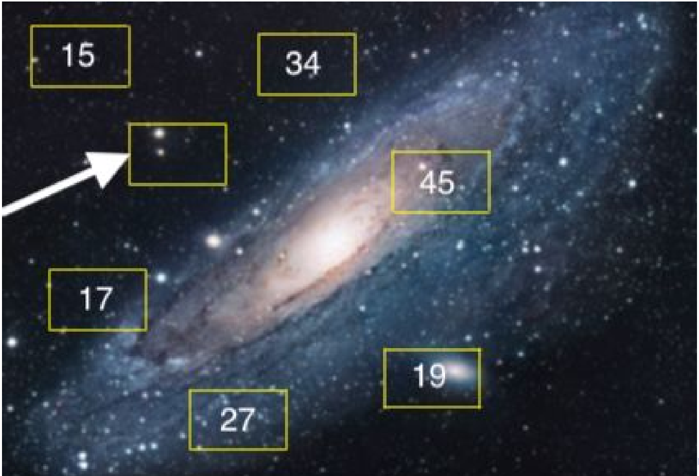
Further issues

- Force some observations, *e.g.*
  - observe area around gravitational wave
  - follow an asteroid



Further issues

- "Expected image interest":
  - can we somehow estimate how much new information a new image will bring about?
  - objective: maximize "total interest" of images collected
  - advantage for a mathematical model here



In summary

- First attempt to model/solve telescope scheduling
- Ongoing work, no definitive results yet
- Methods:
  1. Telescope scheduling as a mathematical optimization problem
  2. Heuristic methods:
     - nearest-neighbor
     - rolling horizon, based on a model for the traveling salesman problem
- Future work:
  - online version (image processing)
  - extend to different objectives
  - deal with real-time constraints
  - exact method?
  - reinforcement learning?