IAC-21-B6.2.8

Towards Explainability of On-board Satellite Scheduling for End User Interactions

Cheyenne Powell(1, a) **and Annalisa Riccardi**(1, b)

(1) *Department of Mechanical and Aerospace Engineering, University of Strathclyde, 75 Montrose Street, Glasgow G1 1XJ, United Kingdom.*
(a) *cheyenne.powell@strath.ac.uk.*
(b) *annalisa.riccardi@strath.ac.uk.*

Abstract

Satellite scheduling is a requirement for automating routines and tasks prior to execution on given satellite/s. Various techniques and tools are used with the optional incorporation of AI depending on the schedule constraints and available resources for memory allocation. Regardless of the technique and approach taken, most autonomous scheduling systems experience challenges enabling an interaction between the user and the system. This affects trust in the system that can lead to manual handling of data that wastes time and resources. Therefore, to reduce these situations from occurring and save costs, the user needs explanations on decisions made autonomously by the system.

An optimal scheduling approach was taken with the use of Constraint Programming (CP) for allocating on-board tasks for a single satellite's schedule. A schedule was derived for short-term planning where tasks were evaluated on duration, cost and resource requirements These results were analysed for their feasibility and optimality; and in doing so, an Computational Argumentation (CA) layer was developed to provide explanations on whether the tasks scheduled, supported, or conflicted with the temporal and/or resource constraints.

To depict the stages and relationships of these internal arguments, an entity relationship graph was created containing the proposed schedule solutions that were evaluated based on their corresponding conflicts/agreements. Due to the nature of these arguments and their respective constraints, another argumentation approach was used to derive basic causalities to provide information on reason of failure and impact on schedule. For end user interactions, the design of explanation layer was investigated allowing the user to select different parts of the proposed schedule enabling a basic output description displayed to assist and enhance the users understanding. This approach will also give the user the possibility to propose changes in the solution and evaluate its feasibility/optimality as well as deriving conflicts with the current schedule. This will allow for growth to build more advanced explainable techniques for sophisticated and complex schedules.

Keywords: Scheduling, Constraint Programming, Explainability, End User Interactions, Argumentation

1 Introduction

Many spacecraft operations were initially manually controlled from Earth before autonomous spacecrafts were developed to introduce on-board scheduling capabilities [1]. This proved necessary due to the majority of satellites life-cycles operating with limited or less frequent contact to it's respective control stations [2]. On-board Satellite Scheduling is a technique used for automatically assign tasks to take place within a given time period while ensuring that these tasks are carried out efficiently and optimally [3]. With this comes high levels of computational resources on demand [4], and to enable a quick response, different techniques are required based on the tasks to be carried out by the satellite/s.

One of these techniques is the Satellite Constellation Scheduling Problem (SCSP), a concept of the Single Machine Problem (SMP) used for coordinating multiple satellites where each satellite has been given their own individual activities to complete [3]. Within SCSP there are various approaches, one of which is the Contention Heuristic, used to prioritise high priority tasks over tasks with lower priority and where there are tasks of the same priority, a level of disagreement is calculated to determine an appropriate solution incorporating temporal constraints [5]. Meta-heuristic is another approach that utilises genetic algorithms that can handle large volumes of solutions, however, require a known feasible solution to be effective [6].

In support of the approaches previously mentioned, different programming methods are used, which includes but are not limited to: offline and online scheduling with CP programming [3][1][4]. Due to the possibilities of increased demands exceeding a satellite's capacity, there is the need for trade-offs to take place within a scheduler with regards to it's relevant constraints. In addition, priority segmentation can be applied to separate tasks into smaller subsets due to the computational costs hence the application of sub-optimal scheduling when deriving large schedules. [7] Depending on the type of satellite, components on board must go through a sequential process such as receiving, and processing data before information can be transmitted to the ground station when communication is established [8]. As a result it is necessary to ensure the schedule is optimal to improve the efficiency of the performance of the satellite.

CP is used for solving several search problems. Within a CP solver, different types of searches can be applied to enhance the performance and attain the most suitable results depending on the problem that's being solved [9]. The CP implementation within OR-Tools, provides the user with different search options namely; Greedy, Guided Local, Tabu etc. with the flexibility of an automatic search where the solver will choose an appropriate one for the given problem [10].

Based on the solution provided, there can be challenges faced by the user in understanding how the solver converged to a certain solution, and as a result an explanation is required.

Explainability AI (XAI) is a technique used for providing information to the user explaining how a solution has been derived by a solver automatically. This can be provided in different ways including graphs, text and images in various combinations depending on the scenario. Explainability can increase trust, transparency and improves the interactions between human and machine [11][12]. Following the explanation to the user, it is ideal to provide the user the opportunity to interact and query the solutions provided, therefore, once a query takes place with the requirement of a response from a machine, results in a computational argument taking place.[13]. Computational argumentation covers different argumentational frameworks, namely, Abstract, Bipolar and Tripolar, which consist of agreements and disagreements and neutral stance on decisions or suggestions made [13]. Therefore, based on the type of queries taking place, the explanation layer will determine the type of argument to be initiated. However, before an explanation or an argument can take place, communication must first be established, requiring a suitable user interface enabling interaction with the system.

This paper focuses on an approach towards explainability from the creation of an offline schedule generated from a manual schedule using a CP solver and the implementation of a computational argumentation layer. It contains 4 sections excluding Introduction and Conclusions.

- Section 2 introduces the satellite scheduling problem.
- Section 3 provides a mathematical formulation for the objective and constraints of the set scheduling problem
- Section 4 explains how the mathematical formulation is used within the system to implement disagreements and agreements relatons between arguments.
- Section 5 explains the methodology adopted to create the schedule from an initial heuristic.
- Section 6 contains the results obtained followed by a discussion of their main findings.

2 Satellite Scheduling

Earth Observation (EO) satellites in a sun-synchronous orbit requires frequent communication to ground stations and based on the type of satellite, several instruments are used for monitoring agriculture, water resources, natural disasters, depleting the available on-board memory until the next possible opportunity to down-link occurs so they can be processed on ground [7]. The data received can attain noise or may be corrupt based on any anomalies such as environment conditions, hardware failures or limited availability of resources, requiring a reschedule to be uploaded to the satellite to compensate for the corrupt or missing data. Thus, time is lost and overbooking can occurred. On-board processing is therefore beneficial to analyse data retrieved from the instruments and removing those that are corrupted and creating a new schedule to preserve the memory.

Figure 1 represents a relationship between an offline and online scheduler where the scheduler on-board will be updated based on the real time input received from the state of instruments and the offline schedule is derived on ground and uploaded as guidance for the online scheduler.

Short term, Mid term and Long term planning are approaches used for creating offline plans for schedules prior to sending the data to the satellite with times ranging from several months to weeks to days respectively [14][15].

To derive a solution to this problem, there needs to be an offline schedule created that is then optimized. Figure 1 represents a relationship between an offline and online scheduler where the scheduler on-board will be updated based on the real time input received from the state of instruments and the offline schedule is derived on ground and uploaded as guidance for the online scheduler.

Considering Sentinel-2A as baseline EO mission, it was examined how on-board scheduling could be performed using CP techniques.

The following data points were collected, over 6 months, using AGI-STK:

- Satellite position (latitude, longitude) for every second
- Day/night regarding satellite position (shade/eclipse times)

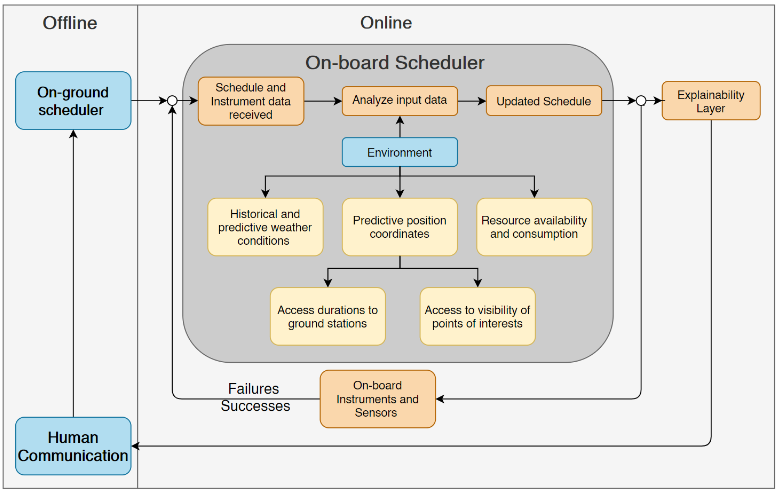


Figure 1: Accepted Schedule

- Duration of all land coverage visible to the satellite's instruments
- Duration of access to ground stations
- For image taking, every channel requires 200MB, with 13 channels this means a maximum of 2.6 GB (all channels combined will be used in this paper)
- Images are taken, during the day, of lands greater than 100 km^2

Based on these coordinates, time intervals for visibility of land, day/night exposures and communication access to ground stations were computed.

Two schedules were calculated, initially a manual schedule implementing a heuristic approach followed by a CP solver approach utilizing the results found from the manual schedule as initial guess for the search. The CP implementation used is the one within Google OR-Tools. Both schedules were generated using the same constraints:

- Only three actions can take place and only one at a time, with an idle time generated when no actions have been executed. These actions are:
  - Take images
  - Process images
  - Down-link images to the ground stations
- Each image requires an estimated memory usage of 2.688 GB that takes approximately 5 seconds to be captured, this duration includes any instrument movement/setup required prior to taking the image.
- The down-link data rate is 280 MB/s.
- An assumed on-board processing rate per image is 50 MB/s.
- The maximum number of images cannot exceed 80% of the satellite's on board total memory.
- Images can only be taken during the day when over land.
- Processing can occur at any time, once at least 1 image has been taken and cannot exceed the number of images taken.
- Down-linking of images cannot exceed images that have been processed and can only take place when there is access to a ground station.
- When down-linking occurs, on-board memory stored is removed from both images taken and images processed.
- At no point can the overall memory be exceeded.

3 Problem definition

The mathematical formulation of the satellite scheduling problem, including all the constraints stated in the previous section, is provided below.
The list of constants for the problem are:

- M_{max}: maximum available memory on-board (set to be 80% of the available memory)
- I_m: memory needed to store an image
- R_m: memory utilised to process (a fraction of) an image in the considered unit of time (5s)
- D_m: memory utilised to down-link (a fraction of) an image in the considered unit of time (5s)

Due to the requirements for calibration of instruments at specific time intervals, it was decided to leave the remaining 20% for calibration, thus allowing calibration to take place at any time.

Noting with A the total number of actions, and with T the scheduling time horizon, the time interval is discretised in 5s intervals and the following variables are defined for the discretisation

- $X \in \{0,1\}^{T\times A}$: binary decision matrix, where if one of its elements, $x_{t,a}$, is equal to 1, it means that the action at index a is taken at the time interval at index t. For notation, a_1 corresponds to the action of taking a picture, a_2 to the action of processing an image and a_3 to the action of down-linking.
- $P \in \mathbb{N}^T$, $D \in \mathbb{R}^T$, $R \in \mathbb{R}^T$: integer auxiliary vectors storing respectively, the total number of images taken, down-linked or processed in each interval in time

$$\begin{aligned} p_i &= \sum_{j=1}^{i} X_{j,a_1} - \sum_{j=1}^{i} X_{j,a_3}\frac{D_m}{I_m} && (1)\\ r_i &= \sum_{j=1}^{i} X_{j,a_2}\frac{R_m}{I_m} - \sum_{j=1}^{i} X_{j,a_3}\frac{D_m}{I_m} && (2)\\ d_i &= \sum_{j=1}^{i} X_{j,a_3}\frac{D_m}{I_m} && (3) \end{aligned}$$

- $M \in \mathbb{N}^T$: integer auxiliary vector representing the memory in use in each interval in time

$$m_i = p_i I_m + r_i I_m, \ \forall i = 1, ..., T \qquad (4)$$

The objective function for the problem is the maximisation of the total number of actions, weighted by the following factors

$$\max\left(\sum_{i=1}^{T} X_{i,a_1} + \sum_{i=1}^{T} X_{i,a_2} + \sum_{i=1}^{T} 2X_{i,a_3}\right) \qquad (5)$$

As outlined in the previous section the following constraints must apply Firstly, only one action at a time can be taken

$$\sum_{a=1}^{A} X_{i,a} \leq 1 \qquad \forall i \in 1, ..., T \qquad (6)$$

The total processed images at any point in time can't exceed the images taken saved in memory

$$r_i \leq p_i \quad \forall i = 1, ..., T \qquad (7)$$

The total down-linked images can occur at any point in time does not exceed the number of processed images

$$d_i \leq r_i \quad \forall i = 1, ..., T \qquad (8)$$

The total sum of memory utilisation at every time interval must not exceed the maximum memory available

$$m_i \leq M_{max} \quad \forall i = 1, ..., T \qquad (9)$$

These constraints have been used in the CP implementation and to assist in the definition of the computational argumentation layer.

4 Explainability

There are several factors than can influence the decisions made for actions to be executed within a satellite schedule (on-board and on-ground). As shown in Figure 1, environmental conditions, satellite positions as well as hardware/component failures can result in the schedule being revised and within the processing itself, can be explained and interpreted as shown in Figure 2.

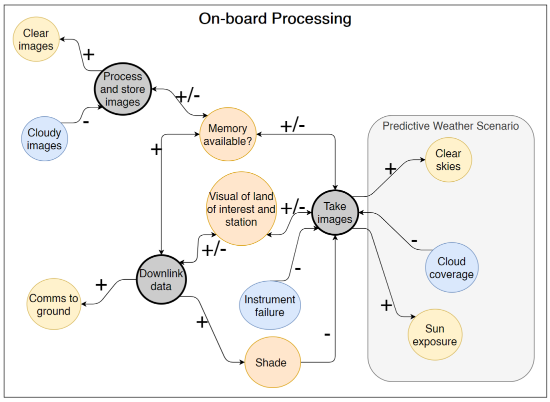


Figure 2: On-board processing

Coloured in yellow $'+'$ indicates agreements with the actions, blue $'-'$ disagreements with the actions to be executed while orange $'+/-'$, indicates an agreement or disagreement depending on the scenario that is occurring at that point in time. This means, if there is sun exposure, clear skies, a visual of land of interest and memory available, this provides a more positive opportunity for the instruments to proceed with image taking. If there is cloud coverage, depending on the percentage, will determine if it is suitable to take the pictures and thus, if instrument failure occurs, this will outweigh all the other conditions and affect the instruments from further taking pictures.

These conditions are contributing factors to the decisions made within the scheduler and are used to infer the scheduler decision making process, and provides explanations to the ground operator.

Computational argumentation is a computational framework that can be used to automatically query and provide feedback to user queries on the solver autonomous decision making. The decisions made by the solver are based on the environmental conditions as well as memory availability as previously shown in Figure 2. Currently, these are based on the defined objective function and constraints, as defined in Section 3.

4.1 Singular Comparative Analysis

Singular comparative analysis is used to compare two schedules to assist with the explanations of the decisions made by the solver. This can be either a comparison between two single decisions taken in a moment in time for the two schedules, or an overall total comparison. Three equations were derived to be used to assist in the generation of the explanations.

- Comparison of total number of images taken

  The overall total sum of images taken in the first schedule, $X^{(1)}$ will be compared to that of the second schedule, $X^{(2)}$. If the total sum of the first schedule is larger than the one of the second schedule, the first schedule would be preferred over the second one and the inequality evaluates to true, false otherwise.

$$\sum_{i=1}^{T} X_{i,a_1}^{(1)} - \sum_{i=1}^{T} X_{i,a_1}^{(2)} \quad \geq \quad 1 \tag{10}$$

- Comparison of schedule idle time

  Idle time is defined as the time when no actions are taking place, as a result, wastes the potential of other activities that could be executing at that time, thus, it is necessary to reduce as much idle time as possible across the entire schedule. Clearly, in the comparison between two schedules, the schedule that has the lower value of idle time will be preferred. If for example, this is the case for the first schedule the inequality below will evaluate to true, false otherwise.

$$\sum_{i=1}^{T} \sum_{a}^{A} X_{i,a}^{(1)} - \sum_{i=1}^{T} \sum_{a}^{A} X_{i,a}^{(2)} \geq 1 \tag{11}$$

- Comparison of total down-link events

  Down-linking is done to send as much data to the ground stations at the window of opportunities and increase memory availability on-board. Therefore, the schedule that performs more down-linking is preferred over the other. If, for example, this is the case for the first schedule the inequality below will evaluate to true, false otherwise.

$$\sum_{i=1}^{T} X_{i,a_3}^{(1)} - \sum_{i=1}^{T} X_{i,a_3}^{(2)} \geq 1 \tag{12}$$

  or equivalently

$$d_T^{(1)} - d_T^{(2)} \geq 1 \tag{13}$$

Overall, these equations were combined to execute a comparison of the schedules to assist with an explanation of their differences.

5 Method

5.1 Manual Schedule Creation

The locations of the satellite were combined and tabulated with the assumption that processing takes place when it is positioned over sea, or over land when in shade. Images are taken while over land and not in shade and down-linking will be prioritised when connections to ground stations are available.

Taking images and down-linking actions were prioritised, with respect to processing and was ensured there were no overlapping of these actions so they can occur sequentially resulting in a heuristic approach taken to calculate memory needed for each action, starting after the first image taken.

Based on the order of the actions, the calculations of memory requirements will occur and once the value approaches its limit, neither image taking or processing actions will execute until a down-link has occurred to free up memory for further images or processing to begin. However, if the memory attempts to decrease to a value below '0', down-linking will no longer take place until an image has been taken and processing of that image is completed. Algorithm 1 provides an example of the approach taken for calculating the memory using such heuristic.

5.2 Schedule created by the solver

Once the manual schedule is generated, it is combined with the environmental conditions (shade, land and ground station visibility intervals) that were created from all the data collected for each day. This was expanded to equal time intervals using the minimum value for the starting point of all daily activities and the maximum time value for the end point of all activities. A boolean encoding was used to encode actions in time, as explained in Section 3. The boolean variables of the potential actions "taking images" were generated from the combination, into a constraint for the solver, of land visibility and

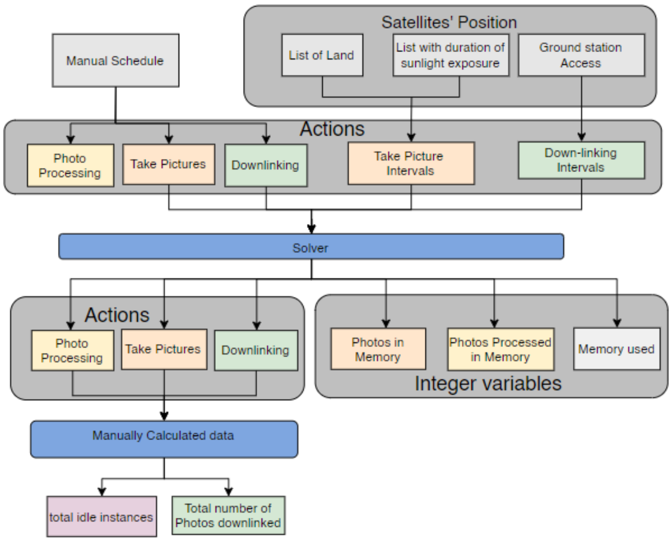


Figure 3: solving process

Algorithm 1 Heuristic approach for memory calculation

while Search through list of actions **do**
  Look for first image to be taken
  if image has already been taken **then**
    calculate how many images are required to be processed
    if images have been processed **then**
      calculate how many down-linking can occur
    else
      look for the next instance for image processing
    end if
  else
    Do not start calculation until image taken
  end if
end while

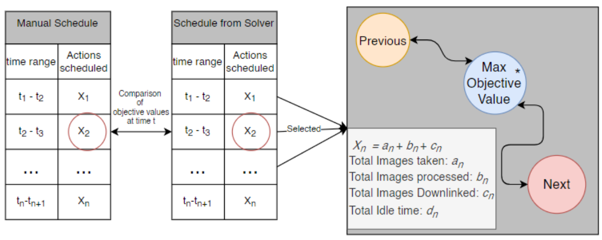
sunlight exposure. These 2 conditions must be "True" for pictures to be taken.

An objective function was created utilizing the maximum sum of the boolean variables within the solver to maximize the occurrences of each action. The total data was processed in sequential batches of 3000 time instances due to resource limitations on the processing computer.

Integer variables: images taken and processed images, were created from their respective boolean variables and used for calculating memory allocation. The integer variable for down-linking was only used for tracking the number of images down-linked and memory freed. Figure 3 represents a summary of the solving process.

5.3 *Argumentation*

After the schedule has been derived, at every action considered, the total values of images taken, processed, and down-linked are immediately compared with the previous values derived within schedule and those in the manual schedule. Figure 4 provides an overview of the processes taking place.

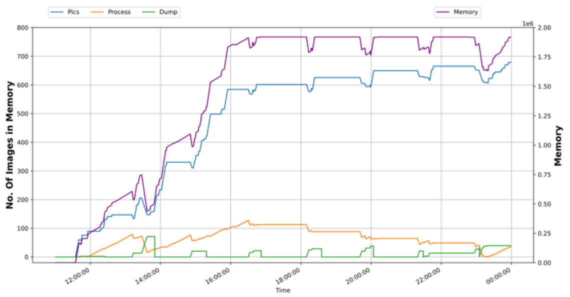


* Refers to Figure 2

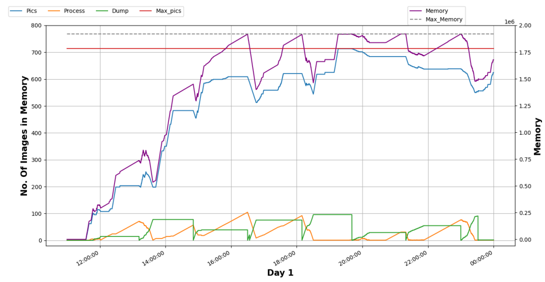
Figure 4: Overall Argumentation Process

Equations derived in Section 4.1 are used to carry out the comparison between the manual and optimal schedules. In Figure 4 the selected option, action x_2 at time t_2, is compared to the values attained in the manual and optimal schedules.

In addition to comparing results of the manual and optimal schedule there is also a comparison of instances of actions against themselves. The previous, current and future actions are considered, based on memory and satellite position, in order to maximize the objective value and therefore used to estimate future decisions.

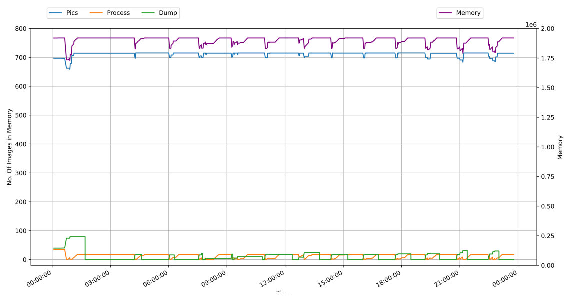


(a) Manual schedule 1

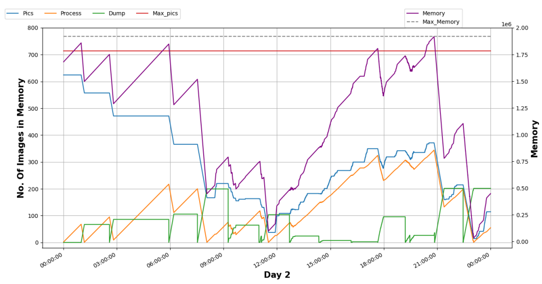


(b) Optimised schedule 1

Figure 5: Day 1 Memory allocation



(a) Manual schedule 2



(b) Optimised schedule 2

Figure 6: Day 2 Memory allocation

Days	Total Pics taken		Total pics processed		Total pics Downloaded		Total Idle Time	
	Man	Solver	Man	Solver	Man	Solver	Man	Solver
1	954	1122	310	500	275	499	3823	1900
2	319	735	284	1256	301	1246	12142	644
3	126	1194	182	1266	187	966	14463	610
4	220	1171	212	1139	207	1265	10039	1412
5	190	1278	192	1160	191	1000	9940	1607
6	198	933	198	1236	198	1166	9174	774
7	218	1200	205	1216	218	1304	9282	497
Total	2225	7633	1583	7773	1577	7446	68863	7444

Table 1: Results for Days 1-7

6 Results

The results were initially generated for Day 1, starting from midday, with the assumption that the memory of each action before execution was 0 GB; this was used as the starting point for schedule creation. The results of the total actions taken across the different days are shown in Table 1, comparing values between the manual and the optimal schedule. Figures 7 and 8 shows the complete manual and optimal schedule for day 1 and 2. The first four rows represent the results from the manual schedule, namely; idle time, take pictures, process images and down-linked. The next three rows represent the night/shade duration, visibility of land and access to ground stations; followed by the last four rows from the generated schedule representing, idle time, images taken, processed images and images down-linked.

Figures 5 and 6 shows the memory utilisation of the memory occupied by the number of images taken, processed and down-linked, at every instance of time. With

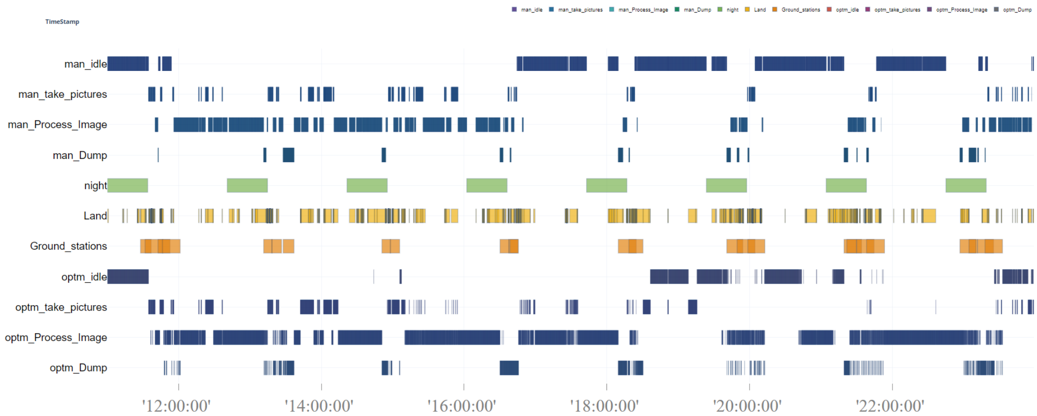


Figure 7: Gantt chart representing both schedules based based on satellites position for Day 1

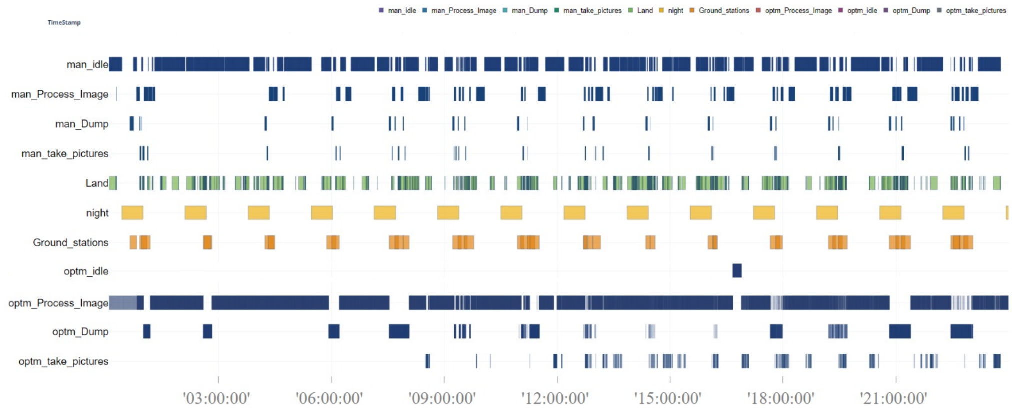


Figure 8: Gantt chart representing both schedules based based on satellites position for Day 2

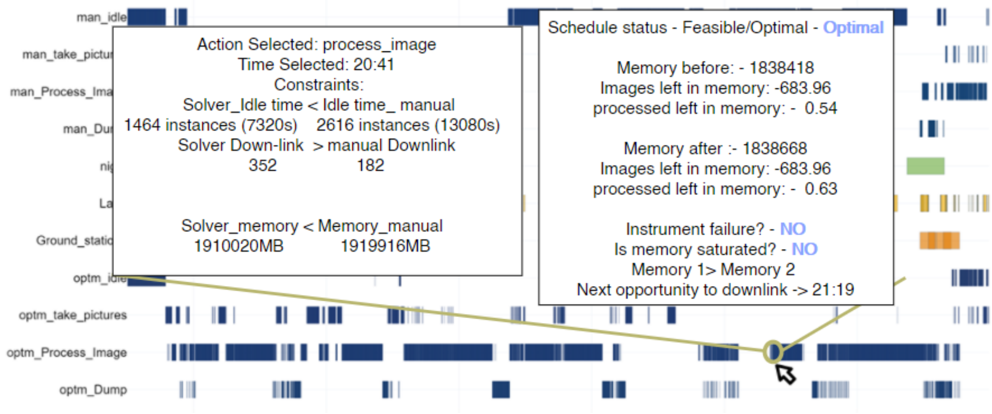


Figure 9: End User Interactions

every processed instance, the memory increases as both processed and non-processed images are kept in memory (in case of processing failure) until down-linking occurs, which reduces the memory and enables the cycle to repeat. The solver tries to manage the on-board memory by replacing image taken with processed ones to ensure down-linking can take place, to preserve memory and increase the value of the objective function.

The totals of the graphs represented in Table 1, show the differences between the generated and the manual schedule over a period of 3 days. The results show that there is a steady increase in the total number of images taken, processed and down-linked between the manual solution and the optimised one, and consequently a reduction of the idle time.

If the user wanted to generate a comparison between manual and optimal schedule, a potential visualisation has been designed. The user is able to select any part of the generated schedule, enabling a simple explanation to appear providing the status at that point along with the comparison between the end users schedule and the generated one. Figure 9 represents an example of how the user can get a visual explaining the status if they selected the option for processing the image. The left field contains a summary of the comparison while the right contains the status of the schedule at that point in time. The information displayed in the explanation box have been obtained with the approach described in Section 4 and 5.1.

7 Conclusion

In this paper, CP techniques were used to generate an offline schedule for short-term planning of an EO satellite in sun synchronous orbit. To create an offline schedule, firstly a manual schedule was created and fed as a first guess to the CP solver. Both the manual schedule and the solver were created by imposing environmental and operational constraints, to calculate the number of images taken, processed and down-linked, to optimise the sum of the overall number of actions taken. Formulations were derived to provide assistance in analyzing the computed results within the system in the form of computational argumentation. In addition, to provide results and encourage an interaction with the user and machine, a display option was explored that would allow the end user to query the system.

Future Works

Different scheduling techniques will be explored to improve offline scheduling results, in terms of optimality and computational efficiency. An adaptive weighted objective function can be defined, depending on the scenarios and status of the on-board memory. Following this, the next stage is to improve explainable approaches for within schedule explainability. Once this has been completed the same approach can be transferred for on board scheduling and explanations. This including explainable techniques for challenges faced on-board such as hardware failures and any anomalies that can occur.

Acknowledgements

This work was supported by ESA contract number 4000132894/20/NL/MH/hm. Thanks to Laurent Perron who provided advice on techniques used for OR-Tools and to Xiang Chen for taking the time to explain and assist with the formulation of the code utilising OR-Tools in making these results possible.

References

[1] I. J. Jeong, G. Papavassilopoulos, and D. S. Bayard, "On-board task scheduling algorithm for spacecraft," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, no. February, pp. 1182–1187, 2000.

[2] F. De Novaes Kucinskis and M. G. V. Ferreira, "An onboard knowledge representation tool for satellite autonomous applications," *Proceedings of the ACM Symposium on Applied Computing*, no. January 2008, pp. 1678–1682, 2008.

[3] K. Sundar, J. Qin, S. Rathinam, L. Ntaimo, S. Darbha, and C. Valicka, "Algorithms for a Satellite Constellation scheduling Problem," *IEEE International Conference on Automation Science and Engineering*, vol. 2016-November, pp. 373–378, 2016.

[4] C. Li, P. De Causmaecker, and Y. Chen, "Data-driven Onboard Scheduling for an Autonomous Observation Satellite," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2018-July, pp. 5773–5774, 2018.

[5] D. Kaslow, "Planning and scheduling of earth observing satellites," *IEEE Aerospace Conference Proceedings*, no. January 2014, 2007.

[6] Benedí C. & Güemes J. and G. dan Sumarno, "Time-window optimization for a constellation of earth observation satellites," *Society*, vol. 3, no. February, pp. 1–8, 2009.

[7] J.C.Pemberton, "Toward scheduling over-constrained remote-sensing satellites," *Proc. of the Second NASA International Workshop on Planning and Scheduling for Space*, pp. 84–89, 2000.

[8] E. Hebrard, M. J. Huguet, D. Veysseire, L. B. Sauvan, and B. Cabon, "Constraint programming for planning test campaigns of communications satellites," *Constraints*, vol. 22, no. 1, pp. 73–89, 2017.

[9] F. Rossi, P. van Beek, and T. Walsh, "Chapter 4 Constraint Programming," *Foundations of Artificial Intelligence*, vol. 3, no. 07, pp. 181–211, 2008.

[10] N. van Omme, L. Perron, and V. Furnon, "or-tools user's manual," *Rapp. tech. Google*, 2014.

[11] D. Khartabil, S. Wells, and J. Kennedy, "Large-scale Argument Visualization (LSAV)," pp. 1–3,

2016. [Online]. Available: http://www.simonwells.org/assets/pdf/khartabil{_}2016{_}euroviz.pdf

[12] M. Fox, D. Long, and D. Magazzeni, "Explainable planning," *arXiv*, pp. 6456–6457, 2017.

[13] O. Cocarascu, A. Rago, and F. Toni, "Explanation via Machine Arguing," pp. 53–84, 2020.

[14] CCSDS, "Mission Planning and Scheduling," no. June, 2018.

[15] S. Peng, H. Chen, C. Du, J. Li, and N. Jing, "Onboard Observation Task Planning for an Autonomous Earth Observation Satellite Using Long Short-Term Memory," *IEEE Access*, vol. 6, pp. 65 118–65 129, 2018.