Global constraints for scheduling data transfer in space missions

Julien Rouzot, Christian Artigues, Philippe Garnier, Emmanuel Hebrard, Pierre Lopez

Global constraints for scheduling data transfer in space missions

J. Rouzot[1,2], C. Artigues[1], P. Garnier[2], E. Hebrard[1], P. Lopez[1]

[1]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse
[2]IRAP, Université de Toulouse, CNRS, UT3, CNES, Toulouse

Abstract

In the context of space missions, where onboard memory resources are limited, efficient data transfer is crucial to avoid buffer overflow. In this paper, we are interested in scheduling data transfer for ESA's Rosetta mission. This problem has been addressed with heuristic methods that show good results, and even optimal results in few cases. In line with a previous work, we propose an exact method to solve this problem. We use constraint programming with two new global constraints. The first constraint links the decisions on priority to the memory usage by reasoning about bandwidth allocation. The second constraint breaks some symmetries of the model. The first results show that the exact method is relevant to solve this data transfer problem. This method aims at being generic, which means that it could be applied for other space missions, or any problem that involves a dense ranking.

Introduction

Rosetta was launched in 2004 by the European Space Agency (ESA) with the goal of performing a detailed study of comet 67P/Churyumov–Gerasimenko, and ended in 2014. In this paper, we address the problem of data dump in the Rosetta mission as defined in (Chien et al. 2015; Rabideau et al. 2017). This problem is closely related to the data dump planning for the Philae lander (Simonin et al. 2015). In both cases, scientific activities produce data that are temporarily stored in the memory buffers of the instruments. The objective here is to minimize the highest peak of memory usage over all buffers during the whole time horizon. In the Rosetta data dump problem, we assume that the experiments are already planned and we must assign a priority for each memory buffer, that will drive bandwidth sharing. A priority ranking is to be computed for each downlink window, that is, the period in which Rosetta can communicate with Earth. This priority ranking is formally a total preorder: every two buffers have either equal priority, or one has priority on the other. Buffers with same priority rank share the bandwidth equally using a Round-Robin algorithm (circular queue). Thus, the buffers with the best rank can use all the initial bandwidth and the other buffers can use the residual bandwidth, until all buffers are empty or there is no more residual bandwidth. We consider that the fill rates of the buffers is known, as they mainly depend on the planning of scientific experiments. The priorities can only be changed at the beginning of each downlink window. As the relationship between the priorities and the evolution of the memory usage during the downlink window is complex, a classical constraint programming (CP) approach is inefficient. To improve the resolution, we have developed the following global constraints:

- `PRIORITY TRANSFER` is used to relate the maximum memory peak among the buffers within a single downlink window, the memory usage for each buffer at the end of the window and the priority ranking. We use the algorithms presented in (Hebrard et al. 2022) to infer new bounds on the priority variables according to the current maximum peak during the whole time horizon.
- `DENSE RANKING` efficiently filters the priority variables domains to ensure that the priority variables follow a *dense ranking*, e.g., (1,2,2,3) for 4 buffers, and thus exclude equivalent solutions, such as (1,3,3,4).

Below, we describe these global constraints and we provide preliminary experimental results assessing this approach. Finally, we present the planned future work to improve this CP model.

overlapping Memory Dumping Problem (oMDP)

In the Rosetta data dumping problem or overlapping Memory Dumping Problem (oMDP) (Rabideau et al. 2016), we consider m downlink windows in which the dump rate δ_j is constant. We have n buffers with a limited capacity. The memory usage of these buffers at the beginning of a downlink window j is denoted by $mem_{i,j}$ while $r_{i,j}$ is the memory peak for buffer i during downlink window j. Our objective is to minimize the worst memory peak during the whole mission: $\min rmax = \max_{i,j} r_{i,j}$.

We define some priority variables $p_{i,j}$ with integer domains from 1 to n. Priority rank 1 corresponds to the highest priority. As the fill rate over time is known, we can compute the memory usage over time for each buffer, for a particular priority assignment, using the `SIMULATION` algorithm described in (Hebrard et al. 2022).

Figure 1 shows the evolution of the memory usages for a simple instance (fill rates in color) according to a given (sub-

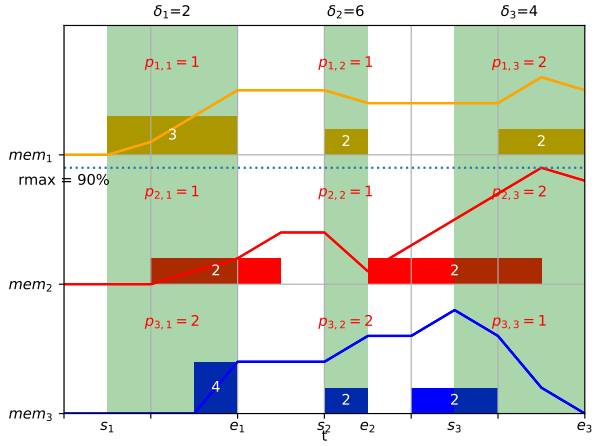

Figure 1: Evolution of the memory usages according to a complete priority assignment.

optimal) solution. During the downlink windows (in green), the data can be dumped at a rate δ_j, and the bandwidth is shared among the buffers according to their priority rank.

Global Constraints

The main reason for oMDP hardness, is the complex relationship between the priority variables and the resulting memory usage during the downlink windows. This complexity motivates the development of the `PRIORITY TRANSFER` global constraint, that links the priority, memory and peak variables ($p_{i,j}$; $mem_{i,j}$; $r_{i,j}$). Moreover, the way we model the priorities as absolute ranks is very straightforward but introduces equivalent solutions. We propose the `DENSE RANKING` global constraint to impose a *dense ranking* on the priority variables to break these symmetries efficiently.

Priority Transfer

When variables $p_{i,j}$ and $mem_{i,j}$ are fixed, we can use the `SIMULATION` algorithm (Hebrard et al. 2022) to efficiently compute $r_{i,j}$ and $mem_{i,j+1}$. Moreover, as our objective is to minimize $rmax$, we can filter the priority variables domains according to $rmax$ current upper bound. Indeed, with the `SINGLE WINDOW` algorithm (Hebrard et al. 2022), we can deduce a partial order on the buffers with respect to a given threshold. Using $rmax$ upper bound as a threshold, we can deduce some priority values to remove from the domains thanks to the resulting partial order.

Dense Ranking

Several assignments of the priority variables yield equivalent priority rankings (e.g., (1,1,2,2,3) and (2,2,4,4,5)). To break this symmetry, we impose that the assignment of the priority variables forms a dense ranking:

- At least one priority variable must take the value 1 (highest priority rank).
- If value i is not assigned, value $i+1$ cannot be assigned.

The `DENSE RANKING` constraint is proposed to efficiently filter the priority variables domains according to these previous rules. The filtering is done via two algorithms:

- `PRIORITY SAT` checks whether there is a valid support (i.e., consistent with the domains). This algorithm takes $O(n\ \log(n))$ to find a support or to prove that none exist.
- `SYMMETRY FILTERING` performs bound consistency, using the previous algorithm as an oracle. This algorithm takes $O(n^3\ \log(n))$ to filter the priority variables.

Preliminary Experimental Results

To evaluate the utility of the `DENSE RANKING` constraint, we generated 30 random instances of different sizes: 2 buffers and 2 windows (*2b/2w*); 4 buffers and 4 windows (*4b/4w*); 6 buffers and 6 windows (*6b/6w*). There is 16 buffers in Rosetta's real instances and the whole mission is split into segments of 64-94 downlink windows. We solve these instances with and without the `DENSE RANKING` constraint with a time limit of 100 seconds and with Google OR-Tools Original CP solver[1] (van Omme, Perron, and Furnon 2014). Experimental results show a clear improvement of the number of instances solved to optimality using the `DENSE RANKING` constraint.

Instance	Dense Ranking	% Optimal Solutions
2b/2w	No	100%
4b/4w	No	40%
6b/6w	No	30%
2b/2w	Yes	100%
4b/4w	Yes	**90%**
6b/6w	Yes	**50%**

Table 1: Percentage of optimal solutions for a 100-second time limit with and without `DENSE RANKING` constraint.

Conclusion

We presented two new global constraints to improve the resolution of overlapping Memory Dumping Problem, and we experimentally showed the efficiency of the `DENSE RANKING` constraint. This constraint is generic and can be used in any problem that requires a *dense ranking*. The `PRIORITY TRANSFER` constraint allows an efficient filtering of the priority variables, but needs upper bounds on $rmax$ to be really effective. Designing heuristics to quickly find good solutions is critical to improving our CP model.

We focus our future work on the integration of heuristics described in (Hebrard et al. 2022) to quickly provide a good upper bound on $rmax$ variable. The impact of these heuristics will be studied with wider experiments (on Rosetta real instances as well as on a large data set generated from these instances). We also plan on improving the filtering algorithm for `DENSE RANKING`.

[1] https://developers.google.com/optimization/cp/original_cp_solver

References

Chien, S.; Rabideau, G.; Tran, D.; Troesch, M.; Doubleday, J.; Nespoli, F.; Ayucar, M. P.; Sitja, M. C.; Vallat, C.; Geiger, B.; et al. 2015. Activity-based scheduling of science campaigns for the Rosetta orbiter. In *Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-15*. Buenos Aires, Argentina.

Hebrard, E.; Artigues, C.; Lopez, P.; Lusson, A.; Chien, S.; Maillard, A.; and Rabideau, G. 2022. An Efficient Approach to Data Transfer Scheduling for Long Range Space Exploration. In Raedt, L. D., ed., *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 4635–4641. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Rabideau, G.; Chien, S.; Galer, M.; Nespoli, F.; and Costa, M. 2017. Managing spacecraft memory buffers with concurrent data collection and downlink. *Journal of Aerospace Information Systems*, 14(12): 637–651.

Rabideau, G.; Chien, S.; Nespoli, F.; and Costa, M. 2016. Managing Spacecraft Memory Buffers with Overlapping Store and Dump Operations. In *Workshop on Scheduling and Planning Applications, International Conference on Automated Planning and Scheduling (SPARK, ICAPS 2016)*. London, UK.

Simonin, G.; Artigues, C.; Hebrard, E.; and Lopez, P. 2015. Scheduling scientific experiments for comet exploration. *Constraints An Int. J.*, 20(1): 77–99.

van Omme, N.; Perron, L.; and Furnon, V. 2014. OR-Tools user's manual. Technical report, Google.