Metaheuristics to design satellite constellation

Enguerran Grandchamp [*] Vincent Charvillat [†]

[*] Alcatel Space Industries
26, Avenue J.F. Champollion, BP 1187,
F-31037 Toulouse cedex, France
Email: enguerran.grandchamp@support-extern.space.alcatel.fr

[†] Enseeiht Informatique
2 rue Charles Camichel, BP 7122,
F-31071 Toulouse cedex 7, France
Email: Vincent.Charvillat@enseeiht.fr

Keywords : tabu search, optimization, satellite constellation design

1 Introduction

At the beginning of the space conquest the space-systems where composed of unique satellite. With the growth of the needs and the explosion of the technologies, the space-systems became more and more complex. Now they are composed of many satellites working together : we talk about constellation. The aim of the constellation design, is to optimize the satellite positions in order to obtain the best performances at the lowest cost. In fact, launching a satellite is expensive and reducing the size of the constellation is necessary.
The field of satellite constellation design is complex because of the size of the exploration space, the presence of local optimum, and the time consuming criterion. For this reason, applying any optimization algorithm over the whole constellation using a black-box criterion leads to an impossible resolution. To bypass these obstacles we decide to split the constellation and the exploration space to simplify the search. We also use approximated criterions to reduce the evaluation of a solution and to analyze its behavior.
The article first presents the terminology and the modelling step of the problem. Secondly we talk about resolution. The last part shows an overview of the results. On conclusion we present the perspectives of the algorithm.

2 Terminology and modelling step

In order to drive the optimization process at different levels we decompose a candidate solution as follow.
A **Satellite** is defined by six **orbital parameters** [12] $Sat = (a, e, i, \omega, \Omega, M)$ which are respectively the semi-major axis, the eccentricity, the inclination, the argument of perigee, the latitude of ascending node, and the mean anomaly.
A **Constellation** is a set of satellites working together (that is the six orbital parameters of each satellite). The size and shape of the constellation depend on the application field and also on the expected performances. In the observation field Spot constellation [10] (figure 1 a) is composed of 3

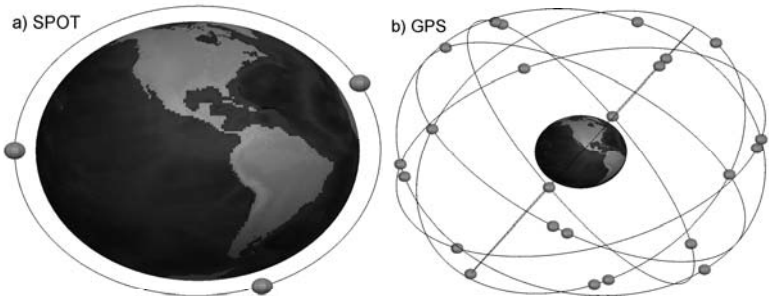

Figure 1: Spot and GPS constellations

satellites in a low orbit ($822Km$) in order to come back over the same area as soon as possible. For a navigation purpose GPS constellation [11] (figure 1 b) is composed of 24 satellites at an elevation of 20200 km.

To sort the different kind of orbit we define the notion of **Orbital Class** which is a set of six **fuzzy intervals** [9] over the six orbital parameters that define a satellite. The intervals are fuzzy because the frontier of many classes are not clearly defined.

The **Orbit Database (ODB)** is a set of n orbital classes $ODB = \{Cl_i\}_{i\in[1,n]}$. Each Cl_i integrates six fuzzy intervals.

We define a **Configuration** of classes as follow : $C_i = (S_{i1}, ..., S_{in}) | \forall k \in [1, n] S_{ik} \geq 0$. Where S_{ik} is the number of satellites of the Cl_k class in the C_i configuration. There is an infinite number of constellation that belong to a same configuration (each parameter set that respect the fuzzy intervals are acceptable).

We associate an evaluation to all of these elements.

Value of a constellation

The optimization process aim to maximize the quality of the constellation. The natural criterion is based on a simulation process. In fact, a propagation scheme of all the satellites on their orbit is realized to evaluate the performance of the constellation.

Let us consider in the rest of the article the following situation : *We want to find the constellation with the minimum number of satellite that satisfy a simple coverage of the earth.* This study case is only proposed to illustrate our framework. Several different situations more complex and challenging are also added in our work.

To evaluate the criterion we have to sample both time and space. The time is a set of p instant $t_j (j \in [1, p])$, the space is a set of q sampled earth areas $A_k (k \in [1, q])$. For each (t_j, A_k) couple we define a local performance P_1 such as $P_1(Const, t_j, A_k) = 1$ if A_k is visible from at least one satellite of the constellation $Const$ at time t_j, 0 otherwise. The value of $Const$ is given by

$$V(Const) = \min_{j\in[1,p]} (\sum_{k=1}^{q} P_1(Const, t_j, A_k)) \quad (1)$$

This evaluation is the most reliable value but it is also the most expensive one (time consuming).

Value of a satellite

We set a value attribute to each satellite Sat belonging to the constellation in order to sort them within the constellation (and to evaluate their contribution to the good or bad performance). This value is directly compute from the simulation process. It leads to increase the ratio between the time spend during the simulation and the information returned. We define P_2 as follow : $P_2(Sat, t_j, A_k) = 1$ if A_k is visible from Sat at time t_j. The value of a satellite is

$$V(Sat) = \sum_{j=1}^{p} \sum_{k=1}^{q} P_2(Sat, t_j, A_k) \quad (2)$$

Value of a configuration
The aim of this evaluation is to sort the configurations to select the one that is able to satisfy the problem with the best performance.
After having explored a configuration $Conf$ (evaluation of many constellations that belong to this configuration) we can give a value to the configuration. In our study case we evaluate a configuration with the following relation

$$V(Conf) = \max_{Const \in \Gamma}(V(Const)) \tag{3}$$

where Γ is the set of visited constellation that match the configuration. The number of visited constellations should be precisely set in order to have a representative value without spending too much time with the simulation.
Value of a class
To find the interesting classes within the ODB we evaluate their **contribution** to the value of the visited configurations. The evaluation of the Cl_j class is given by

$$V(Cl_j) = \frac{1}{N_j} * \sum_{Conf \in \Omega} S_{ij} * V(Conf) \tag{4}$$

where N_j is the number of configurations with $S_{ij} > 0$, Ω is the set of visited configurations.

3 The algorithm

The decomposition of the problem (search space and criterion) leads to a multi-layer algorithm. Because of the different nature of the parameters to optimize we decide to apply different optimization technics. At the lower level we have to optimize continuous parameters (the orbital parameters). We choose a classical analytical algorithm : the **Steepest Descent (SD)**. At a higher level an heuristical split of the exploration space is done using the ODB. At the top level a meta-heuristic algorithm, integrating different memory structures, is used to analyze and improve the search. We use a **Tabu Search** TS with advanced features such as strategic oscillations ([4],[3]).
We detail below both the optimization inside a configuration (see section 3.1) and the search of the best configuration (see section 3.2).

3.1 Continuous optimization within a configuration with SD

In this section, we simplify the problem by considering the configuration as being set. The problem consist of finding the optimal orbital parameters for each satellite within their belonging class.
If the configuration we try to optimize is composed of n satellites we can have to set a maximum of $6n$ parameters. If we apply a typical SD with finite difference algorithm we must evaluate at most 3^{6n} neighbors at each iteration [2]. As the evaluation of a constellation is based on a simulation this step is time consuming and we apply neighborhood reduction to speed up the search.
At the end of this step a value is available for the current constellation (equation 1), for the satellites that compose the constellation (equation 2), and for the current configuration (equation 3). According to this last value we update the classes values (equation 4).

3.2 Configuration setting by standard TS

This level, based on heuristic and meta-heuristic technics is a local search algorithm. The next configuration is choose among the closest neighbors of the current configuration (we can reach a neighbor by adding or removing a satellite, or by changing the belonging class of a satellite).

A typical neighborhood search will evaluate all the neighbors before to choose the new candidate. According to equation 3, the value of a configuration is based on many simulation processes (through equation 1). The size of the neighborhood and the time consuming evaluation of a configuration don't allowed such an evaluation.
To choose the new candidate we prefer a cheaper evaluation which is a kind of estimation based on the previously visited configurations (**short term memory** structure of TS).

$$V_2(C_i) = \frac{1}{Card(\Omega)} * \sum_{C_j \in \Omega} V(C_j) * \frac{M(C_i)}{M(C_j)}, M(C_i) = \frac{1}{Card(C_i)} * \sum_{k=1}^{n} S_{ik} * V(Cl_k) \quad (5)$$

where Ω is the set of already visited configurations.
At each iteration not all neighbors are present in the **candidate list**. In fact, a **tabu attribute** is defined for both satellites and classes in order to disable some transitions. As an example we don't allow to remove satellites recently added (short term memory effect). In the same way classes with worth evaluation are not used to reach the new candidate.

3.3 Guiding the search at a higher level

To drive the TS process to promising area we have to correctly manage the tabu attributes of the classes and the $SD - TS$ trade-off. This management is done using the **long term memory** aspect of TS with the notion of **frequency**. If the use of a class leads to poor configurations we set it permanently tabu. It allows to progressively reduce the configuration space to converge to an optimal subset. This technics, called **strategic oscillations**, is coupled with a transfer of the effort from TS to SD. In order to reduce the computation time, we voluntary reduce the number of iterations of SD at the beginning of the algorithm. But as the algorithm runs, we increase the number of SD iterations to catch better constellations.
The reduction of the configuration space is often linked to the exploration of several areas. During the search we evaluate the progression of the solution. If the nature of the solution (number of satellites, used classes) doesn't change and if the value of the constellation doesn't increase drastically we apply a **diversification process** to guide the search in an other direction.
On contrary, if the quality of the solution increases the **intensification process** is maintain.

4 Application

To have an overview of the performance of each layer we decide to test them separately before to run a complete algorithm. According to the problem defined in section 2 page 2 the optimization criterion is to maximize the number of sample areas covered by the constellation. The optimal constellation is composed of three geostationary satellites uniformly distribute around the equator.

4.1 Optimization inside the known optimal configuration

For each satellite only one parameter (M) is free. For each SD iteration the number of evaluation to find the best direction is $3^{(6-5)*3} = 27$.
The SD converges to the best solution (figure 2 a) in few iterations (cf. figure 2 b). The value of the constellation increase regularly until the optimum value.
We can reduce the size of the neighborhood to accept only one parameter change per SD iteration the number of evaluation becomes $2 * ((6-5) * 3) = 6$. The time spend during the evaluation is reduced by four and the convergence speed is not so affected.
In fact, the gain at the beginning of the descent is higher with the first neighborhood but decreases at the end as if the evaluations are always time consuming. With the second neighborhood the gain is

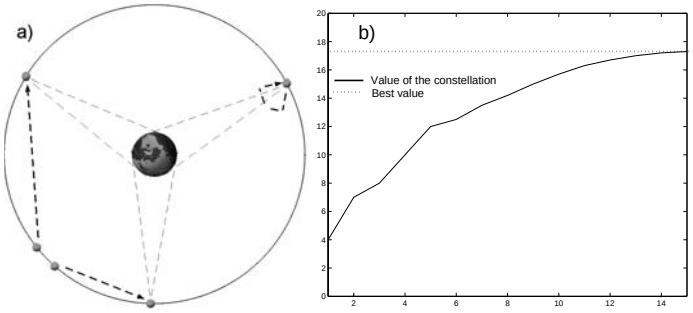

Figure 2: SD Optimization

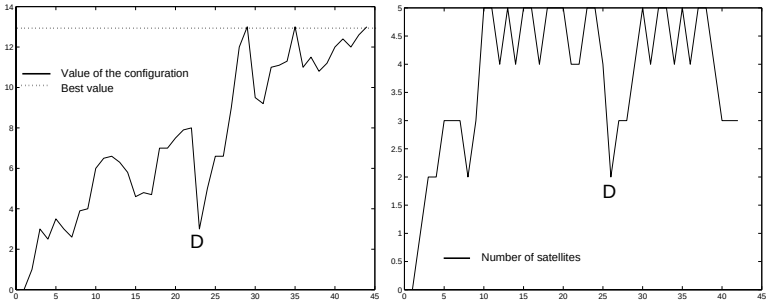

Figure 3: TS Optimization

approximately the same during all the process and the total computation time is lower.

4.2 Optimization of the configuration

To test this layer, we voluntary set a lot of elementary classes. All of them are derived from the Geostationary class by fixing the M parameter. The direct consequence of this choice is to reduce the SD process to the evaluation of one constellation (all parameters are constant). When leaving a configuration we are sure to catch the best constellation.
On the other hand the configuration space and the neighborhood increase significantly. We choose to define 10 classes. The values of M are set in order to have only one best solution (3003 potential configurations and at most 110 configurations in the neighborhood).
For this particular case we can evaluate all the neighbors (equation 3) instead of doing an estimation (equation 5). This run converges to the optimal solution (figure 3) by rejecting classes after a diversification stage (D label on figure 3). At the end of the process the number of visited configurations is less than two percent.

4.3 Optimization from scratch

We now test the whole algorithm. The ODB is composed of three Geostationary classes. For each class the range over the M parameter is $\frac{2*\pi}{3}$ long in order to cover the $[0, 2*\pi]$ range. The best configuration is composed of one satellite of each class with correct M values.
The algorithm converges to the solution in two ways depending on the $SD - TS$ trade-off.
If the number of SD iteration is sufficient, the algorithm converges to the best solution the first time

it visits the best configuration. If the number of SD iteration is insufficient, the best configuration is visited many times before to return the best constellation. But in all cases the best configuration has a good evaluation leading to conserve it at each strategic oscillation step.

5 Conclusion

To reduce the time spend in the algorithm we can act on different parameters. A good use of this algorithm results on a compromise between the time spend in SD and the number of explored configurations. We can reduce the time spend in SD by reducing the size of the classes. But if we don't want to reduce the exploration space (and so potential solutions) we need to add classes. Consequently the complexity of TS increases. The choice of the classes (number, size, nature) is determinant for the algorithm. When exploring the range of the different orbital parameters set, we remark that there is many ways to gather the orbits and several belong to different classes (the following set of orbital parameters $(400km, 0.1, \pi/2, 0, 0, 0)$ belong to the LEO, $Elliptical$ and $Polar$ classes). The cutout and the selection of the classes should be carefully done for each application.
To extend the performances of the algorithm we should include the detection of good structures. When exploring the configurations we attribute a value to the selected classes but the good performances are often due to a combination of different classes. It must be useful to identify such cluster of classes by extending our TS.
When evaluating a class or a configuration, we only compute the financial cost of the constellation from the number of satellites. The precision could be increased by using a more realistic cost that depends on the classes (the cost is not the same for a LEO satellite and for a MEO one).

References

[1] T.A. Ely and al. Satellite constellation design for zonal coverage using genetic algorihms. In *AAS*, 1998.

[2] R. Fletcher. *Practical methods of optimization.* Wiley, 1987.

[3] F. Glover and M. Laguna. *A user's guide to Tabu Search.* Annals of Operations Research, 1992.

[4] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publisher, 1997.

[5] D. G. Goldberg. *Genetic Algorithms in search, optimization and maching learning.* Addison Wesley, 1989.

[6] E. Grandchamp and V. Charvillat. Integrating orbit database and metaheuristics to design satellite constellation. In *Proceedings of ICAI'2000*, pages 733–739, 2000.

[7] E. Grandchamp and V. Charvillat. Satellite constellations optimization with metaheuristics. In *Proceedings of Euro'2000*, page 139, 2000.

[8] G. Dutruel-Lecohier M. Bello Mora, J. Prieto Munoz. Orion - a constellation mission analysis tool. In *International Workshop on mission design and implementation of satellite constellations*, 1997.

[9] Negoita. *Fuzzy systems.* Abaccus Press, 1981.

[10] Spot Image official web site. *http://www.spot.com.*

[11] B.W. Parkinson. *GPS Theory and Applications.* 1996.

[12] O. Zarrouati. *Trajectoires spatiales.* CNES-Cepadues, 1987.