Satellite constellation optimization with metaheuristics

E. Grandchamp[a], V. Charvillat[a]

[a]ENSEEIHT Informatique, 2 rue Charles Camichel,
BP 7122, F-31071 Toulouse cedex 7, FRANCE
Phone:+33 561 588 379 - Fax :+33 561 588 353
E-mail: Enguerran.Grandchamp@enseeiht.fr

ABSTRACT

The method we propose is a new approach to the problem of **satellite constellation design**. The main difficulties of this field are the size of the solution space, the computation time of the criterion and the lack of information to analyse and improve a solution. Our model bypasses some of these obstacles by using an **inverse approach** where services to be fulfilled are highlighted.

Keywords: satellite constellation design, optimization, Tabu Search

ACKNOWLEDGMENTS

This work is financed by Alcatel Space Industries *

1. INTRODUCTION

In the field of satellite constellation design the problem is to find a set of satellites working together to satisfy a certain need. The need is directly linked to the application which could be of various nature like earth observation, telecommunications, data collect, or positioning. A **satellite** is defined by six orbital parameters $(a, e, i, \omega, \Omega, M)^\dagger$ [1] taking their values in a wide continuous range. A **constellation** is a set of satellites.
The classical approach uses a **simulation** to evaluate a constellation. A **sampling** process first provides $N_{instant}$ times and N_{area} interest points on the earth surface. A **propagation** scheme based on orbital parameters provides the positions of each satellite at each time : according to visibility and pose conditions, the **local performances** of the constellation can then be evaluated. The global evaluation (the optimization criterion) finally consists in returning the worst case among the $N_{instant} * N_{area}$ local evaluations (min max type criterion). Some works [2][3] directly apply global (and blind) optimization techniques such as Genetic Algorithms [4] to improve this simulated criterion.
The main drawbacks of such an approach are the size of the exploration space, the time to compute the simulation, the lack of information extracted from the solution evaluation, and the lack of knowledge about the influence of a change on the parameters.
To solve these problems we adopt an "inverse" approach. That is we will construct the solution using a discrete definition of the needs we have to satisfy, we will analyse the solution to learn how changing it to improve the evaluation, and we will use another criterion to avoid the expensive simulation.
In the first part of the article we present the problem modeling. Secondly we deal with resolution and present tabu search and its integration in our method. Some results of the algorithm are shown before conclusion.

2. MODELING

2.1. The service

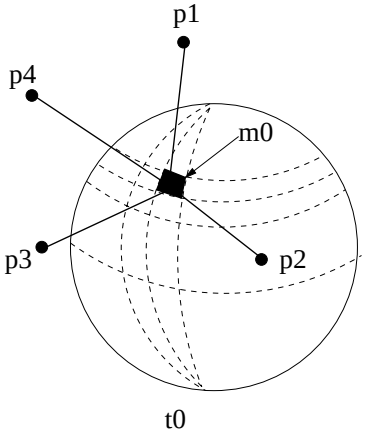
The service stems from the specifications including spatial and temporal informations.
As previously both time and space are sampled. For all (instant t_i, area m_j) samples we postulate one or more ideal satellites positions p_k. A position is the definition of an area in which the satellite should be. The sampling process should be realized for all the problem to obtain a services based structure like :

$$Sv_{total} = \{(t_i, m_j, p_k) : i \in [1, N_{instants}], j \in [1, N_{area}], k \in [1, N_{positions}]\}$$

The specifications could also include different priority levels depending on the sample areas (m_j).

*Alcatel Space Industries, 26, Avenue J.F. Champollion,(BP 1187) F-31037 Toulouse , CEDEX France

†a:semi-major axis,e:excentricity,i:inclination,ω:argument of perige,Ω:longitude of ascending node,M:mean anomalie



Sv=((t0,m0,p1), (t0,m0,p2), (t0,m0,p3), (t0,m0,p4))

2.2. The solutions

A solution is no longer given by the list of orbital parameters of the constellation satellites. We implicitly code an orbit through the list of the services rendered by the corresponding satellite. From this services list Sv_s we can compute (by a robust regression process) the associated orbital parameters P_s : $Sol = \{Sat_s | s \in [1, N_{satellites}]\}$ with $Sat_s = (P_s, Sv_s) | P_s = (a_s, e_s, i_s, \omega_s, \Omega_s, M_s)$. We note N_{max} the maximal number of satellites we can use (extracted from the specifications).
At any iteration the number of satellites $N_{satellites}$ could have any value (eventually nul or greater than N_{max}) but at the end $N_{satellites}$ should be lower or equal to N_{max}.

2.3. The criteria

2.3.1. Last and expensive evaluation

This criterion based on simulation has been previously described. It will be used to evaluate a complete solution but as less as possible because of computation time. It depends on the application we treat : in our study case we choose to design a positionning constellation over a defined area. The local evaluation is the precision of the positionning [5][3] $\epsilon_{t,m}$ at the instant t over the sample area m : $\max_{t \in [1, N_{instants}]} \max_{m \in [1, N_{area}]} \epsilon_{t,m}$.

2.3.2. Cheap evaluation during the algorithm

The previously underlined drawbacks impose to define a **new criterion**. In order to evaluate the current solution we will use the informations about the services contained in the solution coding.
Evaluation of a satellite. The evaluation function of a satellite (Val) is made using the solution coding. We integrate **quantitative contributions** such as the number of services rendered by many satellites (ω_1), the number of services rendered by one satellite ($\omega_2 \geq \omega_1$) and **qualitative contribution** represented by the priority level (ω_3). The ω_3 values are initialized regarding the problem specifications ($\omega_3 = 1$ for each service in abscence of priority). These values will be dynamically modulated during the resolution (see 3.3). Let Sat_i be a satellite satisfying the services $(s_1, ..., s_{n_i})$:

$$Val(Sat_i) = \omega_1 * (\sum_{s_i \in \{Sv_a\}} \omega_3(s_i)) + \omega_2 * (\sum_{s_j \in \{Sv_b\}} \omega_3(s_j)).$$

Sv_a is the subset of services satisfied by Sat_i and others satellites of the constellation, Sv_b is the subset of services satisfied only by Sat_i.
Evaluation of a constellation. We compute the value of a constellation C_j using the evaluation of the satellites it contains. Let N_j be the number of satellites already placed, Sv_s be the set of satisfied services.

$$Ref = \frac{Card(Sv_{total})}{N_{max}}, \quad Val_1(C_j) = \frac{Card(Sv_s)}{N_j}, \quad Val_2(C_j) = \frac{1}{N_j} * \sum_{i=1}^{N_j} (Val(S_i^j))$$

Val_1 is the mean number of services per satellite of the constellation it is used to compare two constellations with the same number of satellites. Ref allows to compare the constellation to the final target : satisfy Sv_{total} services with at most N_{max} satellites. Val_2 is a second evaluation translating the mean "quality" of the satellites.

3. RESOLUTION

The resolution is realized using a multi-levels method. The high level layer is a metaheuristic algorithm using local search and more precisely probabilistic tabu search [9],[10]. At the lowest level we have non-linear parameters estimation and combinatorial choices.

3.1. Local search

The **local search** is based on the exploration of the **neighborhood** $V(Sol_c)$ of the current solution Sol_c to find its successor Sol_+ (as good as possible). The neighborhood of a solution is the set of solutions that could be reached using a defined **transition** : $V(Sol_c) = \{Sol_+ \mid \exists\ T_i\ :\ Sol_c \rightarrow Sol_+\}$.
Let $Sol_c = \{Sat_1, ..., Sat_n\}$ be a n satellites solution. We define three types of transitions :

- T_1 : adding a satellite : $V(Sol_c)_{T_1} = \{Sol_+ | Sol_+ = Sol_c \cup Sat_j, j \notin [1,n]\}$
- T_2 : suppressing a satellite : $V(Sol_c)_{T_2} = \{Sol_+ | Sol_+ = Sol_c \setminus Sat_i, i \in [1,n]\}$
- T_3 : replacing a satellite : $V(Sol_c)_{T_3} = \{Sol_+ | Sol_+ = \{Sol_c \setminus Sat_i\} \cup Sat_j, i \in [1,n], j \notin [1,n]\}$

The neighborhood is composed of the three corresponding subsets $V(Sol_c) = V(Sol_c)_{T_1} \cup V(Sol_c)_{T_2} \cup V(Sol_c)_{T_3}$.

3.2. Tabu search with short term memory

We use Tabu Search (TS) to decrease and strategically adapt the size of the neighborhood to drive the selection of the next solution. TS is a **metaheuristic algorithm** that uses the **history** of the search. We define a new neighborhood for the current solution as $V(Sol_C, H) \subset V(Sol_c)$ where H is the history of the search. The restriction of $V(Sol_c)$ is achieved using three ways. The subsections below details these ways and complete the outlines of our algorithm given in section 4.

3.2.1. Probabilities

The first restriction $V(Sol_c, H)_Q \subset Vol(Sol_c)$ concerns the **probabilistic choice** of the type of transition to be done. In fact this choice is the first to do because of its influence on the shape of the next solution. The probabilities evolve during the algorithm according to H to translate the success or failure of the transitions. The probability P_{T_i} to pull the transition type T_i is : $P_{T_1} = |1 - 2.Q|.Q$, $P_{T_2} = |1 - 2.Q|.(1 - Q)$, $P_{T_3} = 1 - |1 - 2.Q|$ where Q is a quality factor. If we improve the solution using T_1, Q increases to favor the addition of a satellite, if we damage the solution Q decreases to favor the destruction of the current solution. The probabilities are computed to favor the replacement of a satellite when the probabilities of destruction is near the probabilities of construction.

3.2.2. Metaheuristics attributes

To achieved the second restriction $V(Sol_c, H)_{MH-Q} \subset Vol(Sol_c, H)_Q$ we define two **attributes** ($satisfied$, $priority$) for each service and the ($date$) attribute for each satellite. The $date$ is the addition date of the satellite, $satisfied$ is the number of satellites that satisfied the service and $priority$ is the priority level of the service. These attributes are used to set a transition to the tabu status : this transition could not be used to reach the next solution. We declare tabu every T_1 transitions that add lowest priority services or already satisfied services. In the same way we forbid T_2 transitions that suppress recently added satellites. In some cases, the set of non tabu transitions of the selected type could be empty. For this reason we introduce **aspiration criterion** to find a transition. For the T_1 type the selected transition is the one adding less already satisfied services. For the T_2 type the selected transition is the one suppressing the less recently added satellite. The case of T_3 is implicitly treated using T_1 and T_2.
At this step of the restriction we can choose for the transition type T_2 the worst satellite of the constellation according to $Val(Sat_i)$.

3.2.3. Orbit Data Base

The last restriction $V(Sol_c, H)_{ODB-MH-Q} \subset V(Sol_c, H)_{MH-Q}$ only concerns the transition T_1 and is linked to the nature of the problem we deals with. Among the different subsets of services, not all are feasibles within an unique satellite. In fact, both times and positions values contained in the service coding should be compatibles with all the services of the subset (physical and temporal constraints). The **Orbit Data Base** (ODB) which delivers expert knowledge takes a set Sv_1 of services and gives a subset $Sv_2 \subset Sv_1$ of compatibles services associated with a set of orbital parameters which are roughly approximated : $(Sv_2, Ps) = ODB(Sv_1)$. The specific features of the ODB operator are not detailed here. We only want to underline that the subset Sv_2 is produced by ODB by returning the first acceptable one.
After the selection of Sv_2, we have to find the corresponding satellite by estimating the orbital parameters $\hat{P}$ to be fitted with Sv_2. The estimation used is based on **non-linear and robust least square estimation** [7] taking

as initial point the orbital parameters P_s given by ODB. At the end of the process, the set of services Sv_3 really satisfied by the estimated satellite could be different from Sv_2 (One or more services of Sv_2 may be outliers). The difference between Sv_2 and Sv_3 may conduct to the rejection of the proposed Sv_2 subset (for instance in the worst cases if $Sv_2 \cap Sv_3 = \emptyset$). Finally $S_c \cup (Sv_3, \hat{P})$ is the new candidate (S_+).

3.3. Tabu search with long term memory

The long term memory integrates the notion of **frequency**. We memorize the frequency of success ($Freq_{T_1}$, $Freq_{T_2}$, $Freq_{T_3}$) and failure ($Freq^*_{T_1}$, $Freq^*_{T_2}$, $Freq^*_{T_3}$) of each transition type. In order to evaluate how difficult it is to satisfy a service Sv_i we also increment a frequency ($Freq_{Sv_i}$) which memorize the number of failure when trying to integrate Sv_i in the solution (T_1 transition). After a fixed number of iterations N we have to make a decision between Intensification and Diversification strategies. In absence of search progression (computed from the evolutions of Val_1 and Val_2 and the frequencies) we can modify the priority attributes of some services to drive the search in previously unexplored regions. If the frequency based memories identify a service which is hard to satisfy, its priority attribute has to be increased. If the memory detects a repetitive failure of the transitions, the priorities of unsatisfied services have to be increased too. These modifications change the evaluation (see ω_3 in 2.3.2) of the satellites (and so their influence within the constellation) and the neighborhood (and so the evolution of the constellation).

3.4. Tabu search and strategic oscillations

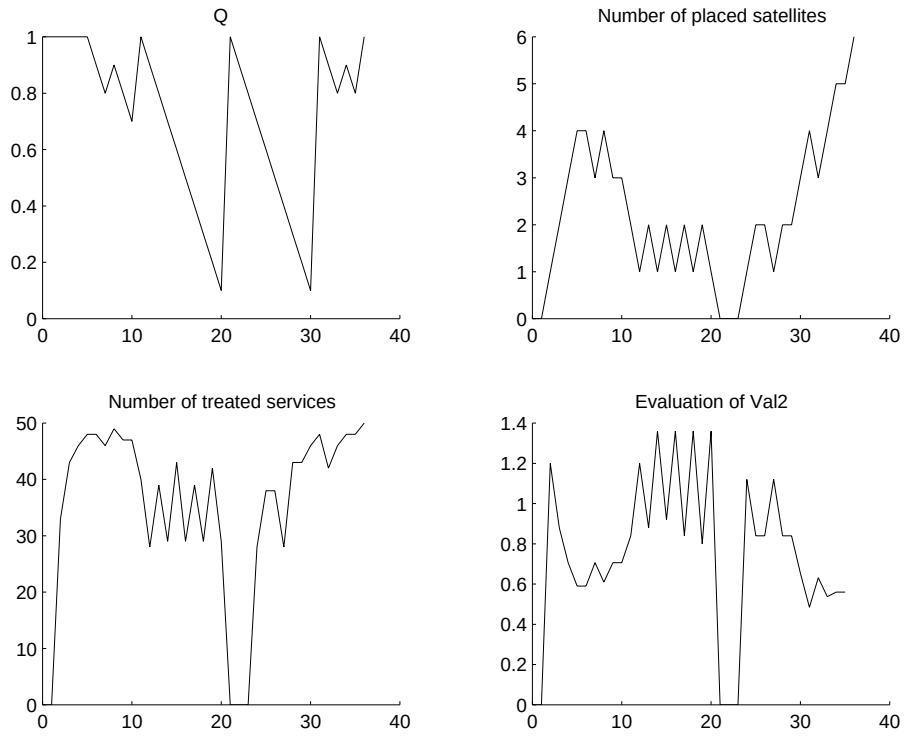
The higher level of the tabu search will be based on the strategic oscillations techniques : the N_{max} parameter will be used as a target and test threshold. A search profile may be chosen in order to begin an easy search from a great N_{max} value to reach smallest values.

4. THE ALGORITHM

Let S_c be the current solution, S^* the best visited solution (according to Val_2 optimization criterion) and C^* its evaluation, H the history of the search and Sv the list of services to satisfy.

Initialization :
  $S_c = \emptyset$. $S^* = S_c$. $C^* = 0$. $H = \emptyset$.
Repeat
  Repeat during N iterations
    Raise a transition type T (probabilities) cf. 3.2.1
    If $T = T_1$ **then**
      Use metaheuristic restrictions, ODB and estimator to reach a new candidate S^+. cf. 3.2.2, 3.2.3.
      Accept or not the new candidate as the current solution ($S_c < -S^+$) cf. 3.2.3.
    elseif $T = T_2$ **then**
      Suppress the worst evaluated satellite of the current solution $\rightarrow S^+$ cf. 2.3.2.
    elseif $T = T_3$ **then**
      Use T_1 and T_2 to reach a new solution S^+.
      Accept or not S^+ according to Val_1 or Val_2 cf. 2.3.2.
    end if
    Update S^*, C^* according to Val_2.
    Update short term memory (probabilities, tabu restrictions) cf. 3.2.
    Update long term memory (frequency based structures) cf. 3.3.
  End
  Decision step about Intensification / diversification balance : modify or not *priority* attributes
    according to the long term memory cf. 3.3
Until end criterion (all services satisfied or maximum number of iterations reached)
If maximum number of iterations is reached **then**
  Return S^* as a good partial solution (could be used as restarting point).
end if

5. APPLICATION



The following example only shows a standard behaviour of our algorithm in order to fulfill 50 services with at most 6 satellites. The proposed solution is the first reached (36 iterations). The plots show :

- the evolution of Q factor which translates a part of the short term memory effect,
- the effect of diversification decision which leads to destroy and rebuild a solution around the 20th iteration,
- the numbers of services and satellites included in the visited solutions,
- the successive values of the Val_2 criterion which points out some good partial solutions (good restarting points).

As a concluding remark some improvements could be introduced to be more opportunist. Just before the 10th iteration we are very close to a feasible solution (48 services,4 satellites) however the search trend is bad and leads to the destruction of the current solution structure : metaheuristics sometimes foil the best plans.

REFERENCES

1. O. Zarrouati, "Trajectoires spatiales",*CNES-Cepadues*,1987.
2. T. Boquillon, E. Lansard "Designing constellation using Genetics Algorithms", *IAF*, 1994.
3. F.Dufour & al, "Constellation Design Optimization With a Dop Based Criterion", *14th International Symposium On Space Flight Dynamics*, 1995.
4. D.G. Goldberg, "Genetic Algorithms In Search, Optimization and Maching Learning", *Addison Wesley*, 1989.
5. B.W. Parkinson, "GPS Theory and Applications Volume 1"
6. P. Axelrad, R.G. Brown, "GPS Navigation Algorithms", *American Institute of Aeronautics and Astronautics*, 1994
7. P.J. Rousseeuw, A.M. Leroy, "Robust Regression and Outlier Detection", *John Wiley and sons*, 1987.
8. International Workshop on Mission Design and Implementation of Satellite Constellations, 1997.
9. F. Glover, M. Laguna, "Tabu Search", *Kluwer Academic Publishers*, 1997.
10. F. Glover, M. Laguna, "A User's Guide to Tabu Search", *Annals of Operations Research*, 1992.