Using genetic algorithms for real time satellite operations scheduling

Giuseppe Corrao, Technological Innovation - Telespazio SpA

Summary Of Proposal

Planning and scheduling are important activities related with every project or activity to carry on, also when they are small and with very low number of resources to allocate.

Even if the two are very often tightly coupled, we can think the planning phase as the definition of the objectives, the available resources and the constraints related to them, whereas the scheduling phase as the assignment of the resources to activities at specific time.

This proposal is relevant with a Genetic Algorithm (GA) based approach to the resource constrained scheduling problem in order to optimize the images acquisition process by an Earth Observation satellite and shows how it can be used in a computer-based application.

Description Of Proposal

In general, scheduling problems are very complex since they are both dynamics (planning can change after new, high priority acquisition requests) and based on incomplete data (in there is no guarantee about the finding of the best solution).

In the most general form the problem can be defined as follow. Given:

- a set of activities (images acquisition) that must be executed,
- a set of resources to perform the activities (ground stations, on-board memory, satellite attitude control etc.),
- a set of constraints which must be satisfied (belonging to different domains as temporal, precedence, availability etc.), and
- a set of objectives with which to judge a schedule's performance (cost function),

what is the best way to assign the resources to the activities at specific times such that all the constraints are satisfied and the best objective measures are produced?

The approach here presented makes use of GA to optimise scheduling problem. GA is an optimization and search technique based on the principles of genetics and natural selection. A GA allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the "fitness" (i.e., minimizes the cost function).

Scheduling problem is here treated in two different phases:

1) Planning

The following entities are defined:

a) *Tasks:* the collection of all requested images listed by priority and request time;

b) *Resources:* each element that can be used in order to accomplish certain task;

c) *Constraints*: relationships among resources that must be accomplished to make a schedule feasible.

All of the three entities contribute in a schedule definition: each task in the schedule may be executed in more than one mode (e.g. it may use different resources or same resources in different times), and each mode may have different resource requirements and constraints.

During the development of the test runs this phase has been performed through STK Engine©, a COM library that uses the well known features of Satellite ToolKit, the state-of-the-art satellite related simulation[1].

2) Search for optimal schedule

This is a stochastic process that uses GA for its purposes. The search phase is carried out defining a population of candidate solutions whose elements are then selected and mated in order to improve their fitness (i.e. minimise the cost function). This phase is the most challenging one, being directly related with the definition an implementation of all the aspects of GA programming:

- Genetic representation of the schedule;
- Natural selection;
- Mating and offspring generation;
- Genetic mutation.

Despite the implemented schema is a straightforward approach to the GA implementation[2] there are at least two noteworthy points:

1) The definition of the cost function. It shall be a multi-objective relationships that aims to represent how well a single schedule is fitting the relevant objective assigning a "fitness" value to each of them. The implementation described in this proposal makes use of a dynamical weighted sum[3], where the weights are modulated on the base of the Pareto ranking[4] techniques.

2) The definition, codification and implementation of the search space and its elements. These elements (also called "chromosomes") shall be coded in optimal way in order to span the entire solution space looking for the best solution (i.e. the one that has the best fitness value). The proposed coding is based on the assumption that each task has an uplink, an access to target and a downlink event whose combination give an unique way (or "mode") to accomplish the task.

This approach aims to solve real-time satellite operations scheduling, even in case of multi-satellite application (e.g. constellation as Cosmo-SkyMed). The cross relations among multi-resource/multi constrained tasks are taken into account using appropriate deconflicting algorithms.

Phases Of Realization

The practical development of the idea is split in the following phases:

1) Evaluation of the effectiveness of the proposed approach: scheduling problems involve non trivial aspects as the definition multi-objective cost function, the allocation of multi-constrained resources and the definition and fine tuning of outline variables;

2) Definition of the operative parameters that really characterise the Earth Observation satellite operations (e.g. satellite slew times, orbital maintenance events etc.);

3) GA implementation into a software system, eventually developing a COTS product.

---

1 See http://www.agi.com/products/STKEngine for more information

2 See Annex 1 for in depth description

3 See Annex 2 for in depth description

4 http://en.wikipedia.org/wiki/Pareto_optimality

Each of the three phases requires different skills (computer science, aerospace engineering, software engineering). In particular, phase 1 is currently under refinement (see section below), phase 2 is going to be executed through the collaboration with the space operation team in Fucino Space Centre and phase 3 requires a pragmatical approach not directly related with the scope of this study, so it will not thoroughly investigated in this proposal, even if several test runs were made coding algorithms in .NET environment.

Status Of The Proposal

At the time of writing the point 1 of previous Section has been carried out with encouraging results[5]. Even if through a prototypal approach, all of the algorithms that compose this innovation have been implemented into a test SW that correctly runs over hundred of iterations, converging towards an optimal schedule.

Benefits Of The Proposal

Since the launch of first Cosmo-SkyMed satellite Telespazio is used to operate Earth Observation satellite, but there is not a specific capability in scheduling algorithms development inside the company. Where the cost reduction is a natural consequence of internal development approach (taking also into account the future operative missions to accomplish), the proposed innovation could also lead in a product/service range expansion (planning/scheduling capability can lead to higher added value in technical bids).

Such capability cannot be found even in the other Finmeccanica companies.

Results Of Prior Research Investigation

Despite GA have been used for scheduling problems resolution due their the stochastic nature, technical literature does not give evidence of previous attempt in the field of real time satellite scheduling. The European network of patent databases (esp@cenet - http://www.espacenet.com/access/index.en.htm) contains just a single patent related with GA usage in scheduling problem, but without giving indication about chromosome encoding and cost function definition as described in Annexes 1 and 2.

Fields Of Foreseen Use

The utilisation of this innovation is related to the satellite operations, even if a certain applicability can be found also in ground segment design to evaluate the best ground stations configuration during preliminary phases (mission definition).

Context And Partnerships

The innovation and its applicability was initially developed and verified by the author. A degree thesis is currently under execution together with Aerospace Engineer department of "La Sapienza" University in Rome in order to go further in the fine tuning of the several coefficients that characterise GA.

---

[5] See Annex 4 for preliminary desccription

Annex 1 – Genetic Algorithms Basis

Genetics Algorithms are population-based, metaheuristic optimization algorithms that use biology-inspired mechanisms like natural selection, crossover, mutation and survival of the fittest in order to refine a set of candidates solution.

The advantage of evolutionary algorithms compared to other optimization methods is their "black box" character that makes only few assumptions about the underlying objective functions. Furthermore, the definition of objective functions usually requires lesser insight to the structure of the problem space than the manual construction of an admissible heuristic, so GA perform consistently well in many different problem categories.

The basic functioning of a traditional GA is quite simple, and can be summarised in the following steps:

1) A population of individuals is randomly generated; each individual (also called "chromosome") shall contain the minimal set of information in order to give a possible solution. Each piece of information within a chromosome is called "gene"; the sum of all possible chromosome is called "genome" and represents the solution's search space;

2) Individual chromosomes are selected from the population on the base of their "fitness", calculated through an appropriate cost function;

3) A mating process is performed among the selected chromosomes by crossing over "genetic material" from two parents to recombine it for new solutions; the generated offspring are going to create a new generation of chromosomes. In this phase is also foreseen a certain mutation rate by offspring to exploit the biological diversity, giving the possibility to escape from local optimal configuration;

4) The process is repeated until some stooping condition is reached.

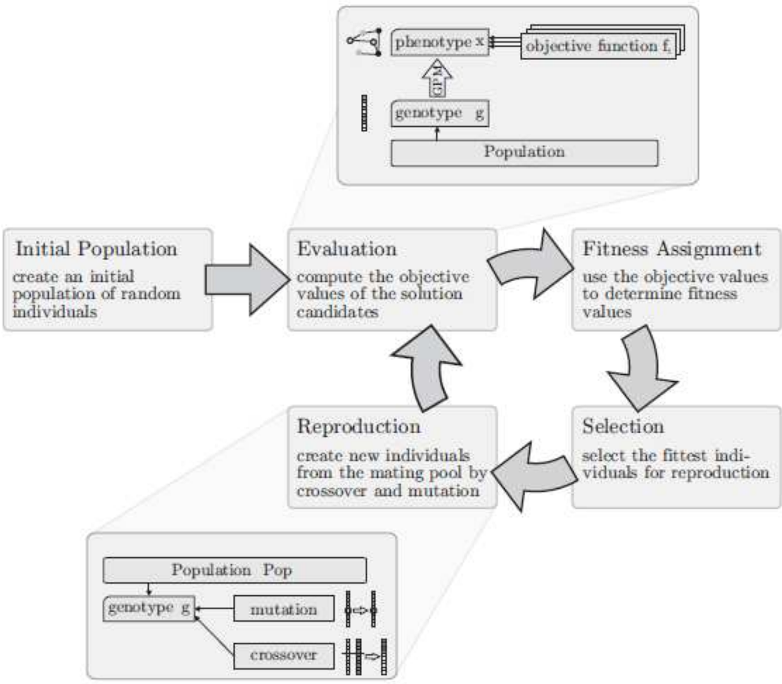


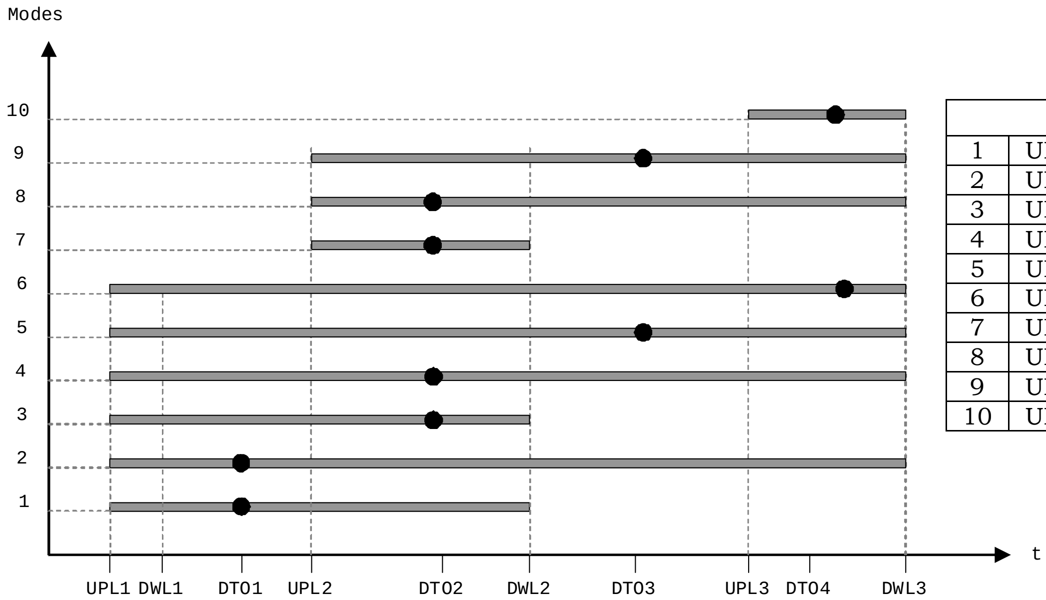
Figure 1 – Evolution mechanism of GA

Annex 2 – Schedule Encoding And Cost Function Definition

Encoding phase is crucial for a correct problem definition, since it shall assure that the entire solution space will be potentially covered by chromosome population.

The basic idea behind the encoding mechanism used is that each task completion (e.g. image available at certain time) requires Upload (UPL), Data Take Opportunity (DTO) and Downlink (DWL) opportunities that are subject to the sequential constraint (UPL- > DTO -> DWL). Each valid sequence UPL-DTO-DWL is called *mode*.

The Figure below shows an example of different modes ($1 \leq m \leq 10$) available to accomplish a certain task, given 3 upload resources (UPL1->UPL3), 3 download resources (DWL1->DWL3) and 4 DTO (DTO1->DTO4).



Modes			
1	UPL1	DTO1	DWL2
2	UPL1	DTO1	DWL3
3	UPL1	DTO2	DWL2
4	UPL1	DTO2	DWL3
5	UPL1	DTO3	DWL3
6	UPL1	DTO4	DWL3
7	UPL2	DTO2	DWL2
8	UPL2	DTO2	DWL3
9	UPL2	DTO3	DWL3
10	UPL3	DTO4	DWL3

Figure 2 – Different modes for a single task

Given *n* task to be accomplished (i.e. *n* images to deliver), each of them having *m* modes of acquisition, we define chromosome the following list of pairs (or *genes*):

T_1	$m_{1,m}$
T_2	$m_{2,m}$
…	…
T_n	$m_{n,m}$

In that way, each chromosome univocally maps a possible schedule, as depicted in the Figure below, that shows an example of schedule with 6 tasks to accomplish:

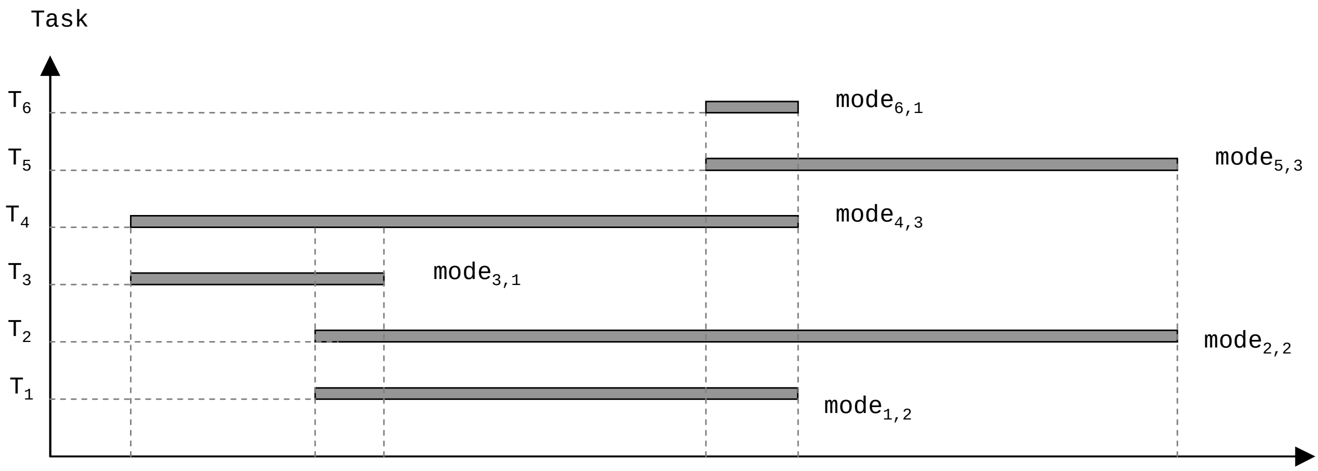


Figure 3 – Schedule representation

Since scheduling is a multi-objective problem, the definition of a cost function for each objective to reach is needed. The objectives taken into account are the following:

1. Minimisation of the Ground Station costs (i.e. the cost related with Ground Station usage);
2. Minimisation of the preset cost (i.e. the time interval between command upload and DTO);
3. Minimisation of the latency cost (i.e. the time interval between DTO and the data download);
4. Minimisation of the access cost (i.e. the time interval between the first potential access and the actual one)

Said n the number of task to accomplish and m the relevant mode index, the fitness value of each chromosome can be calculated using the following cost function:

$$F_{ch} = \left[W_{GS} * \sum_{i=1}^{n} \left(C_{upl,i} + C_{dwl,i} \right) + W_{PR} * \sum_{i=1}^{n} \Delta S_{upl,i}^{m} + W_{LT} * \sum_{i=1}^{n} \left(\Delta S_{dwl,i}^{m} * P_{l,m}^{\Delta S_{dwl,i}} \right) + W_{AC} * \sum_{i=1}^{n} \Delta S_{acc,i}^{m} \right] * R$$

, where:

F_{ch} =Chromosome fitness value

$W_{GS}, W_{PR}, W_{LT}, W_{AC}$ = Cost weights factors

F_{gs} =Ground stations fitness value

F_{pr} =Preset fitness value

F_{lt} =Latency fitness value

i =Number of chromosome's genes

$C_{upl,i}$ =Cost factor of the ith uplink station

$C_{dwl,i}$ =Cost factor of the ith downlink station

$\Delta S_{upl,i}^{m}$ =Time span of the mth uplink opportunity respect to the 1st uplink opportunity

$\Delta S_{dwl,i}^{m}$ =Time span of the mth downlink opportunity respect to the 1st downlink opportunity

$P_{l,i}$ =Priority level of the mth image

R = Chromosome rank value respect to current population

Annex 3 – Genetic Criteria

The following criteria have been implemented in the framework of the proposed algorithm:

1) Number of population's individuals

Its value shall be a trade off between portion of the solution space to be explored at once and processing time requested for each cycle. A population of 32 individuals has been chosen as first guess, even if with low number of tasks to accomplish the population should increase a bit.

2) Stopping condition

The criteria used to decide whether the scheduler should terminate its search or not are:

- After a specified number of iterations,
- when no improvement in the solution quality occurs or
- the process has already yielded a sufficiently good solution.

Once one of the three is reached, the algorithm stops and the best schedule found is proposed as possible solution.

3) Natural selection & recombination

The natural selection process is performed through the assignment of a fitness factor to each individual that compose the current population and then implementing a tournament selection among n participant (randomly chosen among the same population), where the winner is the individual that has the best fitness. Two consecutive winners become parents of an offspring, whose genes are the best corresponding gens of their parents.

4) Elitism

An elitist GA ensures that at least one copy of the best individual of the current generation is propagated on to the next generation. The main advantage of elitism is that its convergence is guaranteed, meaning that once the global optimum has been discovered, the evolutionary algorithm converges to that optimum. On the other hand, the risk of converging to a local optimum is also higher. To take into account pros and cons of elitism, its implementation can be turn on and off during test phases.

5) Mutation

Mutation is an unary search operation that alters a gene of generated offspring with a certain percentage. The scope of mutation is to allow the exploration of random areas of search space, searching for other local minima (or even the global one). Mutation rate shall be a compromise between a too low value (the first minimum found has an high probability to remain the same) and a too high value (in this case GA should become a random search).

A first guess of 0,05 (5%) as mutation rate has been implemented in the test phases.

Annex 4 – Preliminary outcomes

Some test runs have been conducted in order to check the feasibility of the proposed approach. A population of 32 individuals has been implemented to solve a preliminary scheduling problem where 6 images were committed to be taken in a period of 4 days.

The convergence towards a minimum takes only 7 iterations, and the average population fitness vary from 420 (1st generation) to 19 (7th generation):

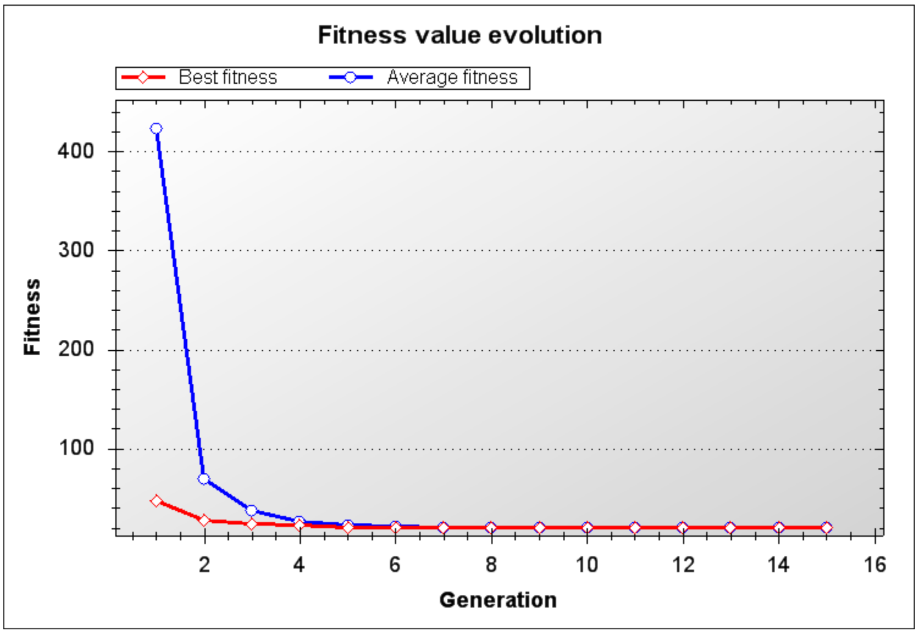


Figure 4 – Convergence towards best fitness value

The relevant schedule is hereafter reported: each task has an upload time and a download time that characterise it length. Within it, an access event is also shown as a red & white dot.

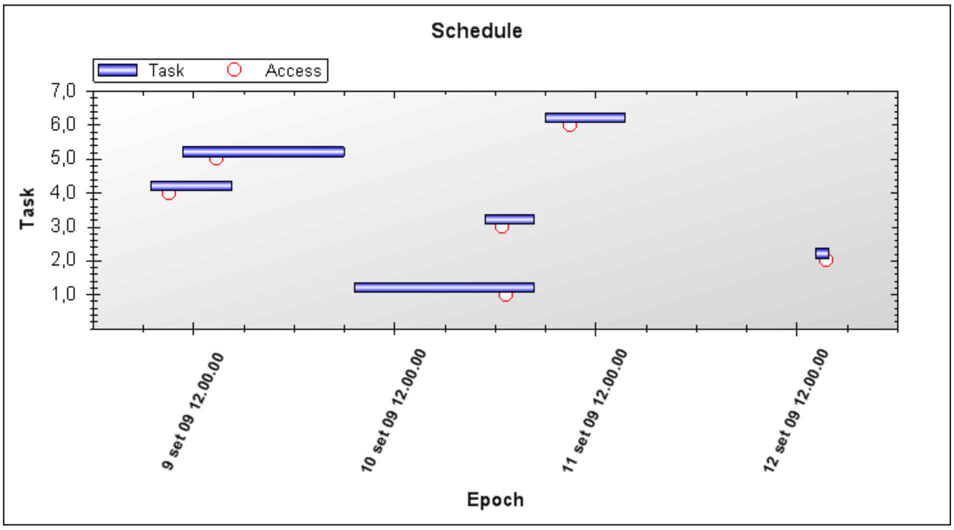


Figure 5 – Final schedule as generated from software test