ENHANCED CONTINUOUS TABU SEARCH IN A HYBRID EVOLUTIONARY ALGORITHM FOR THE OPTIMIZATION OF INTERPLANETARY TRAJECTORIES

Matteo Rosa Sentinella[(1)], **Lorenzo Casalino**[(2)]

[(1)] *Politecnico di Torino, DENER, Corso Duca degli Abruzzi 24, 10129 Torino, Italy, +39 0110904483, matteo.rosasentinella@polito.it*

[(2)] *Politecnico di Torino, DENER, Corso Duca degli Abruzzi 24, 10129 Torino, Italy, +39 0110904453, lorenzo.casalino@polito.it*

ABSTRACT

A hybrid evolutionary algorithm is applied to the optimization of space missions with multiple impulses and gravity assists. The optimization procedure runs three different optimizers, based on genetic algorithms, differential evolution and particle swarm optimization, in parallel; the algorithms are used synergistically by letting the best individuals, found by each algorithm, migrate to the others at prescribed intervals. A mass mutation operator is also employed to diversify the population and avoid premature convergence to suboptimal solutions. A module based on an enhanced continuous tabu search algorithm is introduced in the initialization process to produce a good starting population for the optimization algorithm. The results show the good performance obtained with the hybrid algorithm and the improvement in terms of efficiency and computational cost which is provided, in most cases, by the tabu search initialization process.

1. INTRODUCTION

Evolutionary algorithms (EAs) are optimization procedures which search for the global optimum of a given function in a prescribed search space. Each solution (individual) is represented by the integer or real values of a finite number of variables, which can vary in prescribed intervals. The optimization procedure is started by producing, usually in a random way, an initial population of individuals; then, each algorithm, following its own rules, forces the evolution of the population in order to favor the improvement of the objective function. Some parameters, typical of each algorithm, control the evolution and determine the algorithm capability of finding the optimal solution. Genetic algorithms (GA), differential evolution (DE), particle swarm optimization (PSO) and tabu search (TS) are used in the present article.

In the field of space trajectory optimization, EA were initially employed in conjunction with gradient-based methods [1–4]; GAs have also been used with calculus of variations [5–7] for low thrust trajectories; a combination of artificial neural networks with EAs has been applied to solar sail trajectories [8]. Subsequent works have shown that EAs are well suited to the optimization of impulsive trajectories, which can be described by a limited number of variables. Several studies concerning the optimization of impulsive interplanetary trajectories with gravity assists by means of EAs have been published in the recent literature; tuning of the algorithm parameters and comparisons of the performance of the algorithms have been presented [9–17]. These studies usually agree in stating that EAs are suitable means for the optimization of this kind of missions, even though their conclusions sometimes differ on determining which particular method allows for the best performance. Results may be affected by the choice of the parameters that rule the behavior of the optimization algorithms, and comparison of the results is sometimes difficult due to insufficient details given about the trajectory model.

Evolutionary algorithms are heuristic methods and rely on random processes. Typically, each run may produce a different result, and the performance of such an optimization algorithm can be evaluated

in terms of efficiency (i.e., the probability of finding the global optimum) and number of function evaluations required to attain the optimum. A single EA can be implemented by adopting different strategies and require the specification of a given set of parameters, which control the evolution of the population. The performance of the algorithm is usually dependent on these settings, and the combinations of parameters that produce the best performance (or, at least, acceptable results) may be unknown. An "a posteriori" analysis should be carried out to define the optimal strategies and parameter settings for the given problem. However, there is no guarantee that the same setting is suitable for a different problem.

Hybrid EAs employ different algorithms to deal with the same problem: Information is shared between the algorithms in order to improve the performance. The cooperation between the algorithms is in particular useful when a new problem is dealt with, and "standard" settings, based on the experience of the user and possibly not suited to the problem, are adopted for the algorithm parameters. Thanks to the synergistic effort of different EAs, hybrid algorithms have a larger chance of performing well on different problems, with generic settings of the parameters and without any specific tuning; only population size and number of function evaluations should be varied to take the dimension of the problem (i.e., number of optimization variables) into account.

A hybrid optimization procedure, which runs three different optimizers based on GA, DE, and PSO "in parallel", has been developed [18, 19] and applied to impulsive space trajectories with flybys [20, 21]. The algorithms, which can also be employed separately, are used synergistically, by letting the best individuals found by each algorithm migrate to the others at prescribed intervals. In comparison to existing methods, a "mass mutation" (MM) operator is also introduced in the hybrid algorithm: A high percentage of individuals (typically over 90%) is eliminated when the mean distance between the individuals is smaller than a prescribed value and the optimal objective function is stuck on the same value for more than a prefixed number of iterations [18]. A further improvement of the hybrid algorithm performance is achieved by adding a module based on an enhanced continuous tabu search algorithm in the initialization process. Some individuals of the initial population are not randomly initialized, but are produced by a metaheuristic search realized with the TS algorithm. The use of this procedure to produce a "warm" start for the hybrid algorithm provides, in most cases, improved efficiency and a lower computational cost.

2. EVOLUTIONARY ALGORITHMS

An EA is a population-based metaheuristic optimization algorithm, which borrows its mechanisms from biological evolution: reproduction, mutation, recombination, natural selection and survival of the fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and a cost or merit function determines how the individual is suited to the environment. Evolution of the population takes place with the repeated application of prescribed operators.

In the present paper interplanetary trajectories with a given sequence of planetary encounters are considered; a deep-space impulse may be applied during each interplanetary leg. Each trajectory is defined by the real values of a fixed number of N_p optimization variables (i.e., relevant dates and position, magnitude and orientation of each impulse), which can vary in user-specified ranges and completely determine the trajectory and the corresponding ΔV, which is the function here minimized. For each individual, the mission ΔV is evaluated; a large value is associated to the combinations that do not allow for a solution. The fitness function is then defined as the inverse of ΔV. An initial population of randomly generated candidate solutions usually comprise the first generation. The *extended random initialization* (ERI) procedure of Bramlette [22] is introduced, whereby random initializations are tried for each individual until a significant solution is found or a maximum number of tries (20 in this paper) is reached. This initial effort in terms of computational cost is repaid with a faster convergence to the global optimum, which is reached after a lower number of function evaluations [18]. In the present paper, some individuals may be obtained from the search carried out by employing the

TS algorithm, which is described later. After the initialization phase, the optimization code used here employs GA, DE and PSO in parallel [19], according to the island model [23].

2.1 Genetic Algorithm

GAs [24,25] work on a population of potential solutions applying the principle of survival of the fittest to produce better and better approximations of the optimal solution. A new set of approximations is created at each iteration by selecting parent individuals according to their fitness and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals, which are better suited to their environment than the individuals that they were created from, just as in natural adaptation. At each iteration, the principle of *Elitism* is first used to avoid the loss of good individuals caused by other genetic operators. The best individuals, i.e., those with the highest fitness (lowest ΔV), are stored and replace the worst ones of the following generation; in the present work 3 individuals are selected and saved during each iteration.
The new generation is obtained by first forming a "parent" population, which emphasizes the good solutions and eliminates the bad solutions. The GA optimization tool that has been developed can use three different types of *Selection operator*: Tournament selection, roulette wheel selection and stochastic universal sampling. *Cross-over operator* is then applied to get a new generation of individuals: couples of individuals are randomly selected among the parent population so as each individual is chosen once. Deb's cross-over [26] with either constant ($\eta = 2$) and variable-η is employed to create a pair of new individuals, with the N_p optimization variables obtained from the corresponding values of the two parents. The parameter η controls how close children solutions are with respect to parent solutions (the larger is η, the closer are the solutions). *Mutation* is applied to the new population to increase the number of explored solutions and keep diversity in the population. Some of the variables are changed, according to a small specified probability (2% is used in the present paper; results are only slightly affected by the chosen value); the new variable value is chosen randomly in the specified range. The objective function of each individual of the new generation is finally evaluated; the worst ones are discarded and replaced by the elite individuals of the former generation. The whole procedure is repeated for a fixed number of generations N_g or until a prefixed number of function evaluations is reached.

2.2 Differential evolution

DE [27, 28] is a parallel direct search method which utilizes a population of N_i individuals for N_g generations, just as GA does. ERI can be optionally added as well, to initialize the population. DE generates new vectors of variables by adding the weighted difference between two population vectors to a third one. If the resulting individual exhibits a higher fitness than a predetermined population member, in the next generation the new individual replaces the one it was compared to; otherwise, the old individual is retained. This basic principle can be varied, and in fact there are several practical implementations of DE. Six variants, namely DE/best/1, DE/rand/1, DE/rtb/2, DE/best/2 DE/rand/2, and DE/rtb/1, according to [28], can be used in the developed algorithm. The scaling factor F is chosen randomly between -1 and 1, as suggested by [17]. The cross-over probability [27, 28] is fixed at 0.8.

2.3 Particle swarm optimization

PSO [29, 30] is a population based stochastic optimization technique, inspired by social behavior of bird flocking or fish schooling. The system is initialized with a population of random solutions and searches for the optimum by updating generations. However, unlike GA or DE,in PSO, the potential solutions, called particles, move through the problem space. The values of the N_p optimization variables define the particle position. A velocity vector, which rules the motion of the particle, is also

introduced. The particles fly through the domain by following the one with the best fitness function. PSO is initialized with a group of random particles; each particle moves according to its instantaneous velocity, which is updated at each iteration in order to improve the solution. Two criteria are used: The velocity is changed to move the particle toward the best solution P_{best} that the particle itself has reached in the previous iterations (cognitive acceleration) and toward the best solution G_{best} that any particle in the population has reached (social acceleration). Common, Trelea type 1 and 2 and Clerc's strategies [31] are used in the present algorithm. Particles' velocities are limited to a fixed value V_{max} equal to one quarter of the variable range.

2.4 Enhanced continuous tabu search

Tabu search (TS) is a metaheuristic algorithm originally developed by Glover [32,33], which has been successfully applied to a variety of combinatorial optimization problems. In this paper, an adaptation of TS to continuous optimization problems, called enhanced continuous tabu search (ECTS) [34], which is directly inspired by Glover's approach, is used.

This algorithm starts with a diversification phase, with the aim of finding the most promising areas of the domain. In the original ETCS algorithm, the diversification phase is followed by an intensification phase in order to find the optima in the most promising areas, but in the algorithm presented here this task is fulfilled by the hybrid algorithm with GA, DE and PSO in parallel. At the end of the diversification phase the best results found by the ECTS are inserted into the initial population for the hybrid algorithm.

The diversification phase starts from the best solution *s* among solutions produced randomly ($20N_p$ tries are used here). A set of N_s new solutions is generated during each iteration, dividing the domain in N_s hypercubes, which sorround the current solution *s*, and randomly picking a point in every subdomain. A "tabu" list, which contains solutions belonging to regions of the search domain that have already been explored, is created to avoid the risk of the appearance of a cyclic behavior. The solutions, which are close to solutions in the tabu list, are systematically eliminated. The objective function to be minimized is evaluated for the new solutions and the best one becomes the new current solution, even if the objective function is worse than the previous value. After each 'move', *s* is put into the tabu list. A fixed number of tabu solutions is recorded, and when the tabu list is full, it is updated by removing the first solution entered. Subsequent iterations are performed, and the procedure is repeated by starting from the new current point, until some stopping condition is reached. Usually, the algorithm is terminated after a prefixed number of iterations without any improvement of the objective function.

In the diversification phase ECTS generates a list of "promising" solutions to locate the regions of the search domain which correspond to low values of the function to be minimized. Each current solution s is inserted into the promising list if the objective function is lower than a given threshold (typically, the mean objective function value of all the solution in the promising list). Each solution in the promising list can be considered as the center of a hypersphere called promising area. New solutions are added to the list only if they do not belong to any of the promising hyperspheres. This condition stimulates the search toward new areas of the domain.

In the approach adopted here, which aims at the creations of "seeds" that are provided to the hybrid optimization algorithm, ECTS parameters are adjusted in order to obtain a high degree of diversification in the promising list, since convergence to a possibly suboptimal solution in the proximity of the seeds may occur in the opposite case. Optimization variables are normalized so as they vary between 0 and 1. Hyperspheres of radius $R = 0.1$ are adopted to exclude new solutions in the proximity of solutions in either the tabu or the promising list. At each iteration, N_s hypercubes with geometrical partitioning, as proposed by Siarry and Berthiau [35], are introduced to obtain N_s new solutions; to this purpose, the distances $h_j = R + (R_M - R)/2^{N_s - j}$ are introduced for $j = 1, ..., N_s$, where $R_M = 0.5$ is the maximum allowed distance from the current solution s. The optimization variables of the j-th new solution are chosen randomly (with uniform distribution) in the ranges $[x_s - h_{j+1}, x_s - h_j]$

and $[x_s+h_j, x_s+h_{j+1}]$. The bounds 0 or 1 conveniently replace the interval extreme when the domain limits are exceeded.

2.5 Complete Hybrid Algorithm

The complete hybrid algorithm presented in this paper initially runs the ECTS module. At every iteration, $N_s = 6$ new solutions in 6 concentric hypercubes are generated. Tabu and promising lists with either 10 or 100 individuals have been used in this work, with similar results. For simple problems with limited number of optimization parameters, the 10-individual lists usually provide the same efficiency but require a lower number of function evaluations and 10 individuals are therefore to be preferred; for the most complex problem considered in this paper, 100-individual lists present larger efficiency and similar computational cost, and are adopted here. The diversification phase is stopped after $1000N_p$ iterations or when $50N_p$ iterations have been performed without any addition to the promising list, unless specified otherwise.
The populations for GA, DE and PSO are then randomly initialized, but the promising solutions found by ECTS are inserted into each one. These optimizers are run in parallel according to the island model, performing migration of 5 individuals every 50 iterations. When the aggregation degree, defined as the ratio of the minimum objective function and the mean objective function value of the population, is higher than 0.95 for more than 50 iterations without any improvement, mass mutation is performed, randomly re-initializing 95% of the individuals.

3. MULTIPLE FLYBY MISSION TO SATURN

The trajectory of the Cassini-Huygens mission to Saturn is a typical benchmark for evolutionary optimization algorithms. The spacecraft left Earth in October 1997, and performed a first flyby of Venus after six months in order to increase the trajectory's energy and raise the aphelium. After 14 months another Venus flyby was exploited and then, 4 months later, an Earth flyby put the spacecraft into a trajectory toward Jupiter. In December 2000 a flyby of this planet was performed and the spacecraft reached Saturn in July 2004, with the insertion into an orbit around the planet. The patched-conic approximation is adopted, and the dimensions of the planets' spheres of influence is neglected. The trajectory comprises only ballistic heliocentric legs, which are joined in correspondence of either a planetary flyby or a deep-space impulse.
The trajectory consists of $n = 6$ interplanetary legs. Subscripts $j-1$ and j correspond to the departure and arrival planets of the j-th leg, respectively. The mission ΔV is the sum of the velocity changes which are performed: At Earth departure (ΔV_0); at target arrival (ΔV_n); at the flybys (ΔV_{FBj}, $j = 1, 2, ...n-1$); in correspondence of the deep-space impulses (ΔV_{DSj}, $j = 1, 2, ...n$)

$$\Delta V = \Delta V_0 + \Delta V_n + \sum_{j=1}^{n-1} \Delta V_{FBj} + \sum_{j=1}^{n} \Delta V_{DSj} \tag{1}$$

At departure (subscript 0) the spacecraft leaves a given parking orbit with an impulse at the orbit perigee; in a similar way, the insertion into a prescribed parking orbit around the target planet (subscript n) is performed by means of an impulse at periapsis. One has

$$\Delta V_0 = \sqrt{v_{\infty 0}^2 + 2v_{c0}^2} - v_{p0} \tag{2}$$

$$\Delta V_n = \sqrt{v_{\infty n}^2 + 2v_{cn}^2} - v_{pn} \tag{3}$$

where v_∞ is the relative velocity at the boundary of the sphere of influence (hyperbolic excess velocity), v_c and v_p are the circular and orbital velocity at the periapsis of the parking orbit, respectively. For the present analysis, the spacecraft leaves a 300-km circular low Earth orbit; the flyby height

is constrained above 300 km for all the planets; the final orbit has an altitude of 0.33x150 Saturn radii [17].
At the j-th flyby ($j = 1, 2, ..., n-1$) a minimum height over the planet surface is considered to determine the maximum allowable rotation δ_{max} of the hyperbolic excess velocity v_∞; one has

$$\sin(\delta_{max}/2) = \frac{\mu_j / r_{pj}}{v_\infty^2 + \mu_j / r_{pj}} \tag{4}$$

where μ_j is the planet's gravitational parameter and r_{pj} the radius corresponding to the minimum flyby altitude. The lower value of the hyperbolic excess velocity must be considered when the magnitude of the hyperbolic excess velocity before the flyby $v_{\infty j-}$ differs from the value after the flyby $v_{\infty j+}$. When the maximum rotation is not exceeded the velocity change at the flyby is the change in v_∞ magnitude

$$\Delta V_{FBj} = |v_{\infty j+}| - |v_{\infty j-}| \tag{5}$$

On the other hand, when δ exceeds δ_{max}, the velocity rotation must be partially provided by propulsion with an impulse just before (when $v_{\infty j+} < v_{\infty j-}$) or after (otherwise) the flyby (propulsion is excluded inside the planet's sphere of influence). One has

$$\Delta V_{FBj} = \sqrt{v_{\infty j+}^2 + v_{\infty j-}^2 - 2 v_{\infty j+}^2 v_{\infty j-}^2 \cos(\delta - \delta_{max})} \tag{6}$$

When a deep space impulse is allowed in the j-th interplanetary leg, the corresponding ΔV_{DSj} ($j = 1, 2, ...n$), i.e., the velocity change in correspondence of the deep-space impulse is computed as the vectorial difference of the velocity just before (v_{j-}) and after (v_{j+}) the impulse

$$\Delta V_{DSj} = |v_{j+} - v_{j-}| \tag{7}$$

Initially, a simplified version of the problem is considered, excluding midcourse impulses. When n legs are performed the number of parameters is $n + 1$. For each set of parameters, the optimization procedure solves n Lambert's problems by means of a solver based on Battin's method [36]. When an additional impulse is allowed during each leg, four optimization variables must be introduced for each deep-space impulse. In the present paper, the fraction of time from leg departure to the impulse with respect to the leg time-length and the components of the relative velocity at the leg departure are used [20].
When no midcourse impulse is allowed, the problem has six parameters, namely, the departure time and the time-lengths of five interplanetary legs. The 1997 one-year launch window is here considered; legs concerning Venus and Earth vary between 0.1 to 1.5 years; the Venus-Venus leg varies between 0.5 to 3 times the period of Venus; legs concerning Jupiter and Saturn vary between 0.1 to 0.4 times

Table 1. Mission to Saturn ΔV Requirements

	No deep-space impulse		With deep-space impulse	
Phase	Date	ΔV, km/s	Date	ΔV, km/s
Departure	20/10/1997	5.064	22/10/1997	3.913
I Venus Flyby	22/5/1998	0	1/5/1998	0
DS impulse	-	-	2/12/1998	0.425
II Venus Flyby	26/6/1999	0	1/5/1998	0
Earth Flyby	19/8/1999	0	18/8/1999	0
Jupiter Flyby	13/1/2001	0	13/1/2001	0
Arrival	20/10/2004	0.556	22/10/2004	0.555
Total ΔV		5.621		4.892

the period of the leg outermost planet [17]. A maximum flight time of 7 years is allowed. The function that is minimized is the sum of the ΔV at departure, arrival, and at the flybys. The procedure finds a trajectory with $\Delta V = 5.621$ km/s which uses propulsion just at departure and arrival at target planet (all the flybys are nonpropelled).

The same variables ranges are adopted for the mission with deep-space impulses. In addition, the velocity component, at the start of legs with a deep-space impulse allowed, are let vary between -1 and +2 times the corresponding components of the ballistic trajectory which joins the relevant planets in the prescribed time. When deep space impulses are allowed an improved mission with overall $\Delta V = 4.892$ km/s is found. The resulting optimal trajectory has a single deep space impulse during the Venus-Venus leg and all the flybys are nonpropelled. The characteristics of the missions without deep-space impulses and with a midcourse impulse in the Venus-Venus leg are shown in Table 1 and the corresponding trajectories in Figs. 1 and 2.

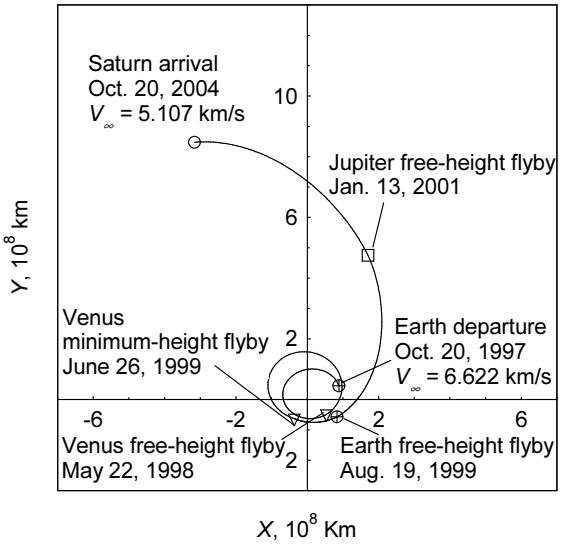

Fig. 1. Optimal Trajectory to Saturn - No deep-space impulse allowed

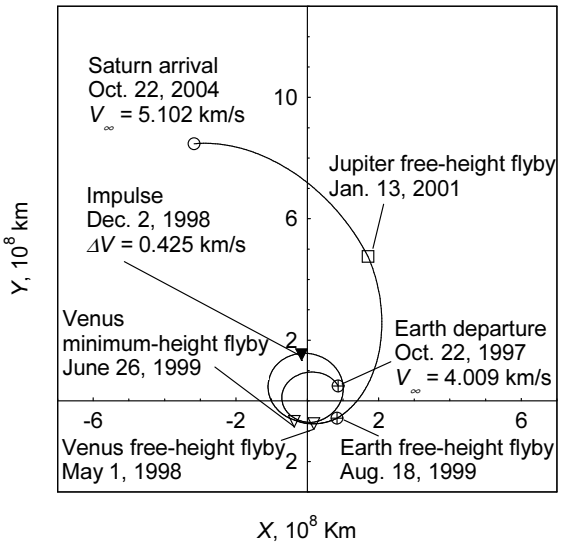

Fig. 2. Optimal Trajectory to Saturn - Deep-space impulses allowed

4. ALGORITHM PERFORMANCE

The performance of the algorithms are evaluated in terms of efficiency (i.e., success rate) and number of functions evaluations required to attain the optimum. An algorithm run is successful and the optimum is considered to be attained when ΔV is within 1% of the global optimum. Each algorithm is run 40 times to obtain an estimation of the success rate, giving an error margin below 10 % with a 95 % confidence (sufficient for a preliminary classification of the algorithms). The simpler problem, which excludes deep-space impulses, has been thoroughly tested in [18] and [19], with slightly different variable ranges and final parking orbit around Saturn (also, efficiency and number of function evaluations were defined slightly differently). The results showed that only DE can reach a 100% efficiency in finding the optimal solution, whereas PSO presents a mean efficiency of 60% in most cases, and GA slightly exceeds 50%. The best strategies for DE is DE/rand/1 with 60 individuals (efficiency $\epsilon = 100\%$); for GA the best result is obtained with tournament selection and Deb's crossover with 100 individuals ($\epsilon = 57.5\%$); for PSO, common strategy achieved $\epsilon = 75\%$ with 30 individuals. These strategies have been used to create the hybrid algorithm, which has been tested by performing multiple runs; each run is stopped after 500000 function evaluations. The results are presented in Table 2, with and without the addition of the diversification phase performed by the ECTS module.
The hybrid algorithm provides a 100% efficiency with a computational cost comparable with the one of DE alone. The ECTS module is not useful, in this case, as it causes a slight increase in number of function evaluations to reach the optimum. The hybrid algorithm is well suited to this problem and the initial computational effort required by the diversification phase (usually around 5000 function evaluations) is not repaid during the following optimization process. If nonoptimal settings are adopted for the hybrid algorithm, (DE/rtb/2 with 60 individuals for DE, $\epsilon = 90\%$, stochastic universal sampling and Deb's Crossover with 100 individuals for GA, $\epsilon = 40\%$, and Trelea's Type I and 24 individuals for PSO, $\epsilon = 60\%$) the results, presented in Table 3, show that the full hybrid algorithm is still able to achieve a 100% efficiency, thus proving that the integrated use of different

Table 2. Performance of the hybrid algorithm with the best set of parameters - No deep-space impulse allowed

	No ECTS	ECTS
Efficiency	100%	100%
NFE min	21133	22975
NFE mean	29704	31658

Table 3. Performance of the hybrid algorithm with nonoptimal set of parameters - No deep-space impulse allowed

	No ECTS	ECTS
Efficiency	100%	100%
NFE min	21320	29774
NFE mean	55240	36866

Table 4. Performance of the hybrid algorithm - One deep-space impulse allowed

	No ECTS	ECTS
Efficiency	92.5%	100%
NFE min	55994	44407
NFE mean	91035	78845

optimisers is synergyc in terms of efficiency with a comparable computational cost. In this case, the addition of the ECTS module remarkably reduces the computational cost, as the average number of function evaluations required to attain the optimum is reduced by one third.
The problem complexity is increased when deep-space impulses are allowed [20]. The performance of the algorithm when a deep-space impulse is allowed only during the Venus-Venus leg ($N_p = 10$ optimization variables) is presented in Table 4, with the settings and strategies of the basic algorithms corresponding to the optimum choice for the nonpropelled case, but the number of individuals of the GA is increased to 200, to account for the larger number of variables. Also, Trelea type 2 strategy is used in the PSO algorithm. Each try is again stopped after 500000 function evaluations. For this problem, the addition of ECTS provides a significant benefit both in terms of efficiency, which reaches 100%, and speed of convergence, with a 15% cut of the required number of function evaluations.
One cannot know a priori how many deep-space impulses are required and in which legs they should occur; a mission, which allows for a deep-space impulse during each leg, has therefore been analyzed to take this consideration into account. The problem has $N_p = 26$ optimization variables and the results are presented in Table 5. The same settings of the one-impulse problem have been adopted, but each try is now stopped after 1500000 function evaluations; also, the tabu and promising lists of the ECTS module comprise 100 individuals. Again, the ECTS module proves to be beneficial in terms of efficiency, raised to 55% from 32.5%, even though a small increase in number of function evaluations is encountered. The ECTS termination criterium of $1000N_p$ iterations seems oversized for the considered problem, as up to 156000 function evaluations may be required during initialization. Tests, also presented in Table 5, have been carried out by reducing the maximum number of iterations to $500N_p$ and $250N_p$ (in the latter case, diversification was also stopped after $25N_p$ iterations without additions to the promising list, instead of $50N_p$). The reduction slightly improved both the efficiency and, as expected, the convergence speed, in particular with the smallest number of iterations for the ECTS module.
The larger number of optimization variables suggests to increase the population size when five deep-space impulses are allowed, and populations of 300, 120 and 30 individuals have also been tested for GA, DE, and PSO, respectively. The algorithm efficiency is improved when this larger population is adopted, as shown in [20], and reaches 53 % when the population is initiated randomly. Results with the addition of the ECTS module, which have been obtained by running each algorithm configuration 100 times to reduce the error margin to 7 % with a 95 % confidence, are shown in Table 6. The benefit of the ECTS-based initialization is even more striking, compared to the previous cases, as the success rate approaches 100% when the initial ECTS diversification is carried out, at the cost of a

Table 5. Performance of the hybrid algorithm (290 individuals) - Five deep-space impulses allowed

	No ECTS	ECTS		
ECTS termination	-	$1000N_p$	$500N_p$	$250N_p$
Efficiency	35%	57.5%	65%	62.5%
NFE min	170291	179223	201444	166509
NFE mean	431494	457302	444453	377800

Table 6. Performance of the hybrid algorithm (450 individuals) - Five deep-space impulses allowed

	No ECTS	ECTS		
ECTS termination	-	$1000N_p$	$500N_p$	$250N_p$
Efficiency	53%	94%	88%	94%
NFE min	242738	259634	267513	230531
NFE mean	392854	436678	441483	444399

small increase in the average number of function evaluations to attain the minimum (the termination criterium of the ECTS module has now a scarce influence on this value).
The beneficial effect of the diversification phase can be understood by considering how the evolution of the population occurs. When a random initialization is adopted, the first-generation solutions have very large ΔV values (typically, more than 30 km/s when none or one deep-space impulse is allowed, above 100 km/s with five deep-space impulses allowed). During the first generations, few good solutions are generated and the population concentrates toward them; this mechanism may favor convergence to a suboptimal solution, when the first good solutions that are generated are far from the region where the global optimum is located. On the other hand, the solutions, which are passed to the hybrid algorithm by the ECTS module, typically present still large, but lower, ΔV values (about 15 km/s when none or one deep-space impulse is allowed, about 30 km/s with five deep-space impulses allowed) and are usually well distrubuted in the search domain. The evolution of the populations is less likely attracted toward a single (possibly suboptimal) region, and the concentration of the solutions occurs only after a larger number of generations, thus increasing the algorithm probability of finding the global optimum.

5. CONCLUSIONS

An hybrid evolutionary algorithm, which uses differential evolution, genetic algorithm and particle swarm optimization in a cooperative way to optimize impulsive spacecraft trajectories has been applied to an interplanetary mission with gravity assists and deep-space impulses. The method shares information between the basic algorithms by letting the best individuals migrate to the other algorithms at prescribed intervals (island model); for this reason, the hybrid algorithm shows better performance in terms of efficiency compared to the basic algorithms, in particular when nonoptimal settings are adopted for the parameters of the algorithms. Therefore, the developed algorithm constitutes a viable means to deal with new problems, when the optimal parameter settings are not known.
A module based on an enhanced continuous tabu search algorithm has been introduced to perform an initialization phase, with the aim of producing a limited set of suitable initial solutions for the hybrid algorithm. This diversification phase is usually beneficial (sometimes remarkably) in terms of algorithm efficiency, and may also reduce the number of function evaluations required to attain the optimum. Tuning of the parameters that control the module may produce further improvements. The use of this complete hybrid algorithm to deal with interplanetary trajectories with gravity assists, characterized by a quite large search space and by the existence of many local optima, has shown that the algorithm usually presents a quite large efficiency in locating the correct global minimum, and that few runs (and few minutes when a standard PC is employed) usually provide the certainty of obtaining the correct solution, with a very low computational cost.

6. REFERENCES

[1] P. Gage, R. Braun, and I. Kroo, "Interplanetary Trajectory Optimization Using a Genetic Algorithm", *Journal of the Astronautical Sciences*, Vol. 43, No. 1, 59–76, 1995.

[2] G. Rauwolf and V. Coverstone-Carroll, "Near-Optimal Low-Thrust Orbit Transfers Generated by a Genetic Algorithm", *Journal of Spacecraft and Rockets*, Vol. 33, No. 6, 859–862, 1996.

[3] G. Rauwolf and V. Coverstone-Carroll, "Near-Optimal Low-Thrust Trajectories via Micro-Genetic Algorithms", *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 1, 196–198, 1997.

[4] J. Hartman, V. Coverstone-Carroll and S. Williams, ”Optimal Interplanetary Spacecraft Trajectories via Pareto Genetic Algorithm”, *Journal of the Astronautical Sciences*, Vol. 46, No. 3, 267–282, 1998.

[5] T. Crain, R. Bishop, W. Fowler, and K. Rock, ”Interplanetary Flyby Mission Optimization Using a Hybrid Globel-Local Search Method”, *Journal of Spacecraft and Rockets*, Vol. 37, No. 4, 468–474, 2000.

[6] B. Woo, V. Coverstone-Carroll and M. Cupples, ”Low-Thrust Trajectory Optimization Procedure for Gravity-Assist, Outer-Planet Missions”, *Journal of Spacecraft and Rockets*, Vol. 43, No. 1, 121–129, 2006.

[7] M. Rosa Sentinella and L. Casalino, ”Genetic Algorithm and Indirect Method Coupling for Low-Thrust Trajectory Optimization”, *42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, Sacramento, CA, Jun. 2006, Paper AIAA 06-4468.

[8] B. Dachwald and B. Wie, ”Solar Sail Kinetic Energy Impactor Trajectory Optimization for an Asteroid-Deflection Mission”, *Journal of Spacecraft and Rockets*, Vol. 44, No. 4, 755–764, 2007.

[9] R. Biesbroek, ”Study of Genetic Algorithm Settings for Trajectory Optimisation”, *54st International Astronautical Congress*, Bremen, Germany, Sep./Oct. 2003, Paper IAF-03-A.P.30.

[10] R. Biesbroek, ”A Comparison of Differential Evolution Method with Genetic Algorithms for Orbit Optimisation”, *57st International Astronautical Congress*, Valencia, Spain, Oct. 2006, Paper IAF-06-C1.4.02.

[11] D.Myatt, V. Becerra, S. Nasuto, and J. Bishop, ”Advanced Global Optimization Tools for Mission Analysis and Design”, Final Report of ESA Ariadna ITT AO4532/18138/04/NL/MV, Call 03/4101, ESA, 2004.

[12] P. Di Lizia and G. Radice, ”Advanced Global Optimization Tools for Mission Analysis and Design”, Final Report of ESA Ariadna ITT AO4532/18139/04/NL/MV, Call 03/4101, ESA, 2004.

[13] M. Vasile and P. De Pascale, ”Preliminary Design of Multiple Gravity-Assist Trajectories”, *Journal of Spacecraft and Rockets*, Vol. 43, No. 4, 794–805, 2006.

[14] C. Bessette and D. Spencer, ”Optimal Space Trajectory Design: A Heuristic-Based Approach”, *Advances in the Astronautical Sciences*, Vol. 124, Univelt Inc., San Diego, CA, 2006, 1611–1628, Paper AAS 06-197.

[15] D. Izzo, V. Becerra, D. Myatt, S. Nasuto, and J. Bishop, ”Search Space Pruning and Global Optimization of Multiple Gravity Assist Spacecraft Trajectories”, *Journal of Global Optimization*, Vol. 38, No. 2, 283–296, 2007.

[16] T. Vinko, D. Izzo and C. Bombardelli, ”Benchmarking Different Global Optimisation Techniques for Preliminary Space Trajectory Design”, *J58th International Astronautical Congress*, Hyderabad, India, Oct. 2007, Paper IAC-07-A1.3.01.

[17] A. Olds, C. Kluever and M. Cupples, ”Interplanetary Mission Design Using Differential Evolution”, *Journal of Spacecraft and Rockets*, Vol. 44, No. 5, 1060–1070, 2007.

[18] M. Rosa Sentinella, ”Comparison and Integrated Use of Differential Evolution and Genetic Algorithms for Space Trajectory Optimisation”, *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, IEEE Press, Singapore, 973–978, 2007.

[19] M. Rosa Sentinella, *Development Of New Procedures And Hybrid Algorithms For Space Trajectories Optimisation*, Ph.D. thesis, Politecnico di Torino, Turin, Italy, Apr. 2008.

[20] M. Rosa Sentinella, L. Casalino, "Hybrid Evolutionary Algorithm for the Optimization of Interplanetary Trajectories," *Journal of Spacecraft and Rockets*, Vol. 46, No. 2, 365-372, 2009.

[21] M. Rosa Sentinella, L. Casalino, "Cooperative Evolutionary Algorithm for Space Trajectory Optimization", *Celestial Mechanics and Dynamical Astronomy*, 2009. DOI 10.1007/s10569-009-9223-4.

[22] M. Bramlette, "Initialization, Mutation, and Selection Methods in Genetic Algorithms for Function Optimization", *Proceedings of the Fourth International Conference on Genetic Algorithms*, R. K. Belew and L. B. Booker (Eds.), Morgan Kaufmann, San Mateo, CA, 100–107, 1991.

[23] Whitley, D., Rana, S., and Heckendorn, R.B., "Exploiting Separability in Search: The Island Model Genetic Algorithm," *Journal of Computing and Information Technology*, Vol. 7, No. 1, 33 47, 1999. (Special Issue on Evolutionary Computing).

[24] D. Goldberg, *Genetic Algorithms in Engineering Design*, Wiley, New York, NY, 1997.

[25] D. Goldberg, and K. Deb, "A Comparison of Selection Schemes Used in Genetic Algorithms", *Foundations of Genetic Algorithms*, G. Rawlins (Ed.), Vol. 1, Morgan Kaufmann, San Francisco, CA, 450–457, 1991.

[26] K. Deb, "An Introduction to Genetic Algorithms," *Sādhanā Journal*, Vol. 24, No. 4, 293–315, 1999.

[27] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optization over Continuos Spaces", ICSI TR–95–012, ICSI, 1995.

[28] R. Storn, "On the Usage of Differential Evolution for Function Optimization", *1996 Biennial Conference of the North American Fuzzy Information Processing Society*, NAFIPS, Berkeley, 519–523, 1996.

[29] J. Kennedy and R. Eberhart, "Particle Swarm Optimisation", *Proceedings of the IEEE International Conference on Neural Networks*, IEEE, Piscataway, NJ, 1942–1948, 1995.

[30] R. Eberhart and J. Kennedy, "A New Optimizer Using Particle Swarm Theory", *Sixth International Synposium on Micro Machine and Human Science*, IEEE, Piscataway, NJ, 39–43, 1995.

[31] I. C. Trelea, "The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection", *Information Processing Letters*, 317–325 ,2003.

[32] F. Glover, "Tabu Search: Part I", *ORSA Journal on Computing 1*, 190–206, 1989.

[33] F. Glover, "Tabu Search: Part II", *ORSA Journal on Computing 2* , 4–32, 1990.

[34] R. Chelouah and P. Siarry, "Tabu Search Applied to Global Optimization", *European Journal of Operational Research*, 256–270, 2000.

[35] P. Siarry and G. Berthiau, "Fitting of Tabu Search to Optimize Functions of Continuous Variables", *International Journal for Numerical Methods in Engineering*, 2449–2457, 1997.

[36] R. H. Battin, "An Elegant Lambert Algorithm," *Journal of Guidance, Control, and Dynamics*, Vol. 7, No. 6, 662-670, 1984.

◄ BACK TO SESSION DETAILED CONTENTS ◄ BACK TO HIGHER LEVEL CONTENTS