Celestial Mech Dyn Astr (2007) 97:73–86
DOI 10.1007/s10569-006-9052-7

ORIGINAL ARTICLE

Evolutionary principles applied to mission planning problems

Bruce A. Conway · Christian M. Chilan · Bradley J. Wall

Received: 8 December 2005 / Revised: 20 July 2006 /
Accepted: 9 October 2006 / Published online: 8 December 2006

Abstract The space mission planning process is considered as a hybrid optimal control problem. Hybrid optimal control problems are problems that include categorical variables in the problem formulation. For example, an interplanetary trajectory may consist of a sequence of low thrust arcs, impulses and planetary flybys. However, for each choice of the structure of the mission, for example, for a particular choice of the number of planetary flybys to be used, there is a corresponding optimal trajectory. It is not a priori clear which structure will yield the most efficient mission. In this work we present a mathematical framework for describing such problems and solution methods for the hybrid optimal control problem based on evolutionary principles that have the potential for being a robust solver of such problems. As an example, the methods are used to find the optimal choice of three asteroids to visit in sequence, out of a set of eight candidate asteroids, in order to minimize the fuel required.

Keywords Mission planning · Trajectory optimization · Evolutionary methods · Genetic algorithms

1 Introduction

Many interesting problems in space mission planning are hybrid optimal control problems. Hybrid optimal control problems (HOCP) are problems that include both continuous-valued variables such as those for position, velocity and mass of the spacecraft and categorical variables in the problem formulation. For the types of problems envisioned here the categorical variables will specify the structure or sequence of events that qualitatively describes the trajectory or mission. For example, for an

B. A. Conway (✉) · C. M. Chilan · B. J. Wall
Department of Aerospace Engineering,
University of Illinois at Urbana,
306 Talbot Laboratory, Urbana,
IL 61801, USA
e-mail: bconway@uiuc.edu

interplanetary spacecraft trajectory a mission could be described by the following sequence of categorical variables: Earth departure, low-thrust powered heliocentric flight, Mercury arrival. An equally valid and perhaps lower cost sequence might be: Earth departure, low-thrust powered heliocentric flight, Venus gravity assist, low-thrust powered heliocentric flight, Mercury arrival.

An approach to the solution of the HOCP is to separate the problem into an "outer-loop" that specifies the structure or sequence of categorical variables that will describe the trajectory or mission (Ross and D'Souza 2005; von Stryk and Glocker 2001) and an "inner-loop" that optimizes the trajectory for the given sequence, e.g. by a two-point-boundary-value-problem (TPBVP) solver or by recent methods that convert the continuous problem into a discrete problem with nonlinear constraints, such as collocation + nonlinear programming (NLP) (Enright and Conway 1992; Herman and Conway 1996). The mission event sequence can always be represented as a sequence of numbers that can be written as a (perhaps lengthy) binary number. It is thus an integer programming problem.

For every feasible sequence of the categorical variables an optimal trajectory can in principle be found. The value of the objective function for that particular mission, i.e. the "cost" of that sequence, can then be associated with that sequence. For "small" problems one obvious approach is to enumerate all of the possible ways in which the mission can be accomplished, determine the optimal trajectory for each of them, and then choose the trajectory with the lowest cost. This would perhaps be a feasible approach for the Earth to Mercury flight, if there were only a few options available such as the intermediate flyby of Venus and/or an impulsive velocity change. However, for problems of even moderate size this is an unreasonable approach since the number of possible sequences increases rapidly as the number of events allowed in a sequence (N_S) increases.

There are additional difficulties, among which are:

(i) the number of events in the sequence of optimal events (N_S^*) may not be known a priori,
(ii) transitions from event A to event B may not be allowed, while going from event A to event C is allowed,
(iii) some events (such as an impulsive departure impulse or an impulsive maneuver for capture into orbit about a target planet) must only be the first or last event in the sequence.

A mathematical framework for describing such problems has only recently been presented. Only a small number of solutions for such problems are available in the literature. For example, Buss et al. (2002) solve an example problem of three robotic arms cooperatively transporting an object from an initial position to a goal. von Stryk and Glocker (2001) create a "benchmark" HOCP they call the motorized traveling salesman problem. It is similar to the well known traveling salesman problem but here the salesman drives a car with limited acceleration/deceleration capability and a limited turning rate. The hybrid problem is to select the order in which a given set of cities should be visited, i.e. the "tour", with each city being visited only once, returning to the departure point, and then, for a given tour, to find the optimal control time history (and resulting path) to minimize the travel time. The problem is simple in concept but for N cities there are N!/2 tours, so the problem is NP-complete. Such problems can be formulated as mixed-integer optimal control problems (MIOCP's), however, no general solution techniques are currently available for such problems.

Thus the "best" approach, or even a feasible approach, for solving all but the simplest problems remains to be determined.

In this work two methods based on an evolutionary principle, in this case a genetic algorithm (GA), are applied to the solution of an example mission planning problem. In the first solution a GA is used as a solver for the "outer-loop" problem of determining the optimal sequence of events constituting the mission. This is combined with a robust "inner-loop" solver that discretizes the general continuous trajectory optimization problem and solves it as a NLP problem. In the second solution of the same problem the "branch and bound" (B&B) method is used as the outer-loop solver and a GA is used to find the optimal trajectories after the continuous problem has been converted into a discrete sequence of Lambert problems.

As an example, the problem of departing Earth orbit and visiting 3 asteroids out of a population of 8 asteroids in various orbits about the Sun is presented. Impulsive velocity changes are allowed at Earth, at the first asteroid flyby (to arrange interception of the second asteroid) and at the second flyby (to arrange interception of the third asteroid). The objective is to minimize the sum of the three velocity changes. Of course no asteroid may be visited more than once. The problem is inspired by the "benchmark" motorized traveling salesman problem but is more ambitious because (i) the optimizer may choose not only the order of interception but which subset of targets to visit, (ii) the system equations of motion are nonlinear, and (iii) the asteroid targets are moving while the cities are of course stationary. The evolutionary methods we have developed solve this problem very efficiently, that is they require evaluation of the cost or "fitness" for a relatively small number of the total number of feasible trajectories.

2 Mathematical description of the HOCP

The first challenge is to create a mathematical formalism for description of the problem. von Stryk and Glocker (2000) have done this as have Ross and D'Souza (2005). The latter approach is better suited to aerospace trajectory/mission planning; it is more flexible, accommodating an event sequence of arbitrary length, and it introduces a method for categorizing unallowed event transitions that is well suited to the solution method we propose to use.

A maneuver automaton can be described in a directed graph or "digraph" (Ross and D'Souza 2005). Figure 1 shows a digraph for an example problem in which there are three possible events that can be combined in some order to qualitatively describe the mission plan. The categorical state space, that is, the totality of events, is graphically depicted, as are the allowed transitions. The edges shown represent the allowed transitions. Depending on the nature of events the edges may require changes in the system dynamics from one vector field to another.

A succinct description of the mathematical formalism for the hybrid optimal control problem follows. Suppose that the categorical state space for the problem (e.g. from its digraph) is

$$Q = \{q_a, q_b, q_c\} \tag{1}$$

with cardinality N_Q ($N_Q = 3$ for this example).

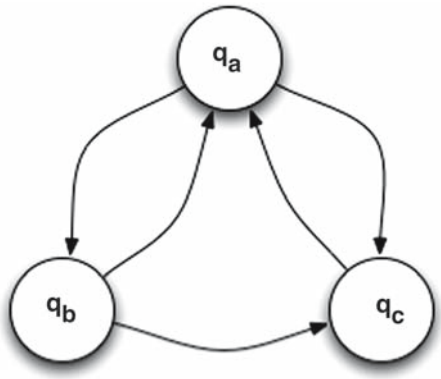

Fig. 1 Digraph for a simple mission

A mission plan is described qualitatively by an event (or maneuver) sequence string, e.g.

$$q_1 = (q_a, q_b, q_c, q_a) \tag{2}$$

with events chosen from the categorical state space.

Associated with each state $q \in Q$ is a continuous-time dynamical system

$$\dot{x} = f(x, u, v, t) \tag{3}$$

with states $x \in R^{N_x^q}$ and controls $u \in R^{N_u^q}$, i.e. the number of state and control variables may depend on the event q, as mentioned previously.

It is important to model or encode the information contained in the digraph, especially the unallowed transitions. Ross and D'Souza (2005) describe a "switching set" $S(q_i, q_k)$. If $S(q_i, q_k) \neq 0$ then transition from event q_i to event q_k is allowed, i.e. (q_i, q_k) is an edge of the digraph whose vertices are q_i and q_k. If $S(q_i, q_k) = 0$ then the transition is not allowed. The system adjacency matrix **A** can be used to encode the digraph [1].

Let matrix **A**, having dimension $N_\mathrm{Q} \times N_\mathrm{Q}$, have elements A_ik such that:

$$A_{ik} = \left\{ \begin{array}{ll} 1, & \text{if } S(q_i, q_k) \neq 0 \\ 0, & \text{if } S(q_i, q_k) = 0 \end{array} \right\} \tag{4}$$

Then **A** contains all the information on the allowed transitions.

The outer-loop problem of determining the optimal sequence (of length N_S, not known a priori) of the categorical variables, q, can now be redefined as the problem of finding the optimal value of the matrix Δ, with $\Delta \in D^{N_\mathrm{Q} \times N_\mathrm{S}}$. That is, the optimal sequence q, of length N_S, can be determined as the "product" of the categorical state space, expressed as a $1 \times N_\mathrm{Q}$, row vector, and the matrix Δ:

$$q = [Q]^* \, \Delta \tag{5}$$

To prevent q from containing unallowed transitions, the Δ matrix must be consistent with the digraph. This requires that elements of Δ satisfy the relation:

$$\Delta_{i,j+1} \in \{A_{ki}, 0\} \text{ for } \Delta_{kj} = 1, \; i = 1, 2, \ldots, N_\mathrm{Q}; \; j = 1, 2, \ldots, N_\mathrm{S} \tag{6}$$

i.e. if $\Delta_{kj} = 1$, then $\Delta_{i,j+1}$ can only assume the values 0 or the A_{ki} element of the adjacency matrix, or to put it another way, in a feasible choice of the Δ matrix, if element $\Delta_{kj} = 1$, then element $\Delta_{i,j+1}$ can be 1 only if the A_{ki} element of the adjacency matrix is 1, that is, if the transition $S(q_i, q_k) \neq 0$.

2.1 Solution of the inner-loop optimal control problem

To determine the *optimal* event sequence q^* one must be able to determine the cost or objective function of the optimal trajectory corresponding to sequence q. That is, one must solve the inner-loop optimal control problem, which will normally be a continuous optimal control problem that can be modeled as a Bolza or Mayer problem for which the necessary conditions of optimality form a two-point-boundary-value problem. There are many approaches available for solving such problems. The solution methods are generally categorized as indirect or direct depending on whether the necessary conditions are or are not explicitly considered, respectively. Practically speaking, indirect solutions employ the Lagrange multipliers or costate variables of the problem while direct solutions do not. Direct methods, e.g. collocation with nonlinear programming (DCNLP) (Enright and Conway 1991; Enright and Conway 1992; Herman and Conway 1996), pseudospectral methods (Ross and Fahroo 2004), or Runge–Kutta (R–K) parallel shooting (Enright 1991; Enright and Conway 1992) have been developed in the past two decades and are probably the most efficient and robust methods extant for the solution of such optimal control problems.

The method used in the solution of the example problem in this work is the R–K parallel shooting method. In this method a continuous optimal control problem is transcribed into a discrete problem, which becomes a NLP problem. The structure of the transcribed problem is shown in the cartoon of Fig. 2. The cartoon shows how continuous time is divided into N segments, the boundaries being the problem "nodes." A representative state time history is shown; note that the state assumes values only at the nodes. Of course for a feasible solution there must be some way of relating the values of the state and control variables at adjacent nodes so that the system differential equations (3) are satisfied. The R–K parallel shooting method accomplishes this. It is illustrated in Fig. 3, which shows just one of the segments from Fig. 2 further divided into 3 internal segments.

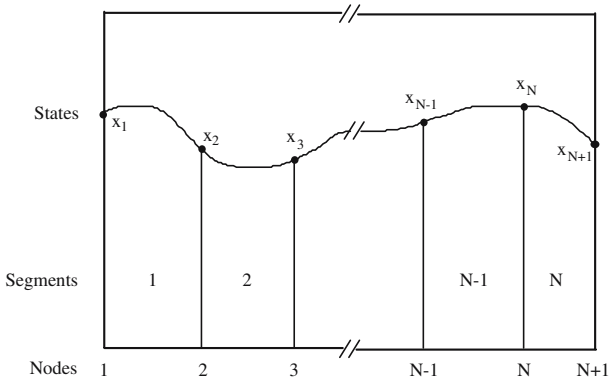

Fig. 2 Discretization of the problem when using collocation or R–K parallel shooting

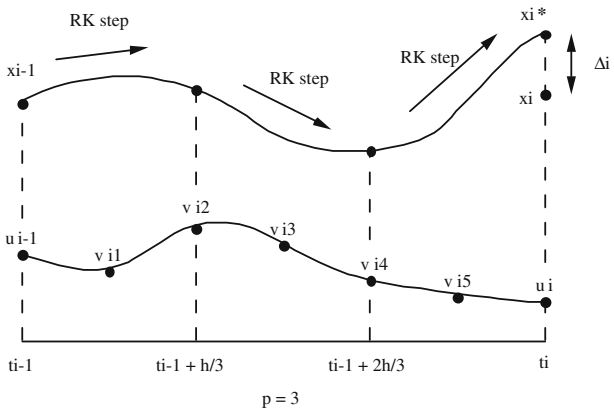

Fig. 3 Structure for 3-step R–K implicit integration scheme

In this scheme, the left side state of each segment is propagated to the right side node by using the control parameters, the u's and v's shown in the cartoon, defined within the segment, yielding a continuity constraint, or "defect." The defect is the difference, at the right-hand side, between the value of the state (x_i^*) obtained by propagation from the left-hand side, using the starting value x_{i-1} and three successive applications of the well known 4-step implicit R–K integration formula, and the current value of the state at the right-hand side known to the NLP solver, i.e. x_i. The defect $x_i^* - x_i$ is a nonlinear constraint equation which is a function of the state vector at the left-hand side and all of the intervening control variables (if any).

The inner-loop optimal control problem thus becomes a nonlinear programming problem. The NLP parameters can be defined as a single vector P that collects all the independent variables, which are ordinarily the discrete state and control variables but often also important event times, such as the final time t_f, i.e.

$$P^T = \left[Z^T\right]$$

where, for example, one might have $Z^T = \left(x_1^T, u_1^T, x_2^T, u_2^T, \ldots, x_{N+1}^T, u_{N+1}^T, t_\mathrm{f}\right)$.

In the same manner, the nonlinear constraints, all or most of which are the defect equations, can be collected into a vector C^T. The NLP problem becomes:

$$\text{Minimize } \phi(P)$$

subject to

$$b_\mathrm{L} \leq \left\{ \begin{array}{c} P \\ AP \\ C\,(P) \end{array} \right\} \leq b_\mathrm{U}$$

where AP is formed by all the linear constraints of the problem, and vectors b_L and b_U are the lower and upper bounds, respectively, of the parameters and constraints.

Fixed initial and terminal conditions for some state parameters are easily enforced by setting the upper and lower bounds to the specified value. The upper and lower bounds for the nonlinear constraints corresponding to the defects are set to zero,

because this forces the solver to choose values for the parameters that satisfy the EOMs when they are integrated forward using the R–K rule within each segment. Once the NLP problem is clearly defined, it can be solved by using dense or sparse solvers such as NPSOL or SNOPT (Gill et al. 1998), respectively. However, since the problem is sparse large problems benefit greatly from using the SNOPT solver.

2.2 Summary of the HOCP

Assuming that there exists an inner-loop solver capable of determining the optimal trajectory for each feasible sequence q, the problem of determining the optimal event sequence q^* (and with it the optimal trajectory and optimal control) becomes an integer programming problem:

$$\begin{array}{ll} \text{Find} & \Delta, N_S \\ \text{subject to} & \Delta \in D^{N_Q \times N_S} \\ & N_S \leq N_{S,\max} \end{array} \tag{7}$$

That is, Δ and N_S implicitly determine the optimal event sequence q^* through Eq. 5.

Returning to the outer-loop optimal control problem; in principle all feasible sequences of events q (of length $N_{S,\max}$) could be enumerated and for each the inner-loop problem solved. The optimal choice would then be the one with the smallest objective function. This total enumeration approach would be costly and very inefficient for all but the simplest problems. The research challenge is to find the optimal Δ and N_S while determining as few as possible of the optimal trajectories for the feasible sequences, i.e. while solving the time-consuming inner-loop problem as few times as possible.

3 Evolutionary optimization methods applied to the HOCP

A genetic algorithm is a search algorithm using the mechanism of natural genetics to find a set of parameters having the best "fitness." The mechanism of a simple GA is described here in accordance with a text by Goldberg (1989). A numerical analysis code used in this research, which is provided by Carroll (1998), is also consistent with this description of GA.

First, the approximate optimal trajectory needs to be determinable given a set of discrete parameters, for example, initial unspecified state and costate variables and an estimate of the final time (if the final time is "free"). This set of parameters needs to be expressed in binary form. The binary numbers are then placed in a "string," becoming an individual. A simple GA searches for the best individual from a population. Each individual in a population is randomly provided in a search area at the beginning of the simple GA operation. The simple GA has better convergence characteristics as the population size, n, is larger and length of string, l, is generally smaller. A simple GA improves the individuals in a population via three genetic operators, reproduction, crossover and mutation, in a generation. Reproduction is a process to select the individuals surviving into the next generation. A tournament selection is the most popular reproduction operator; it compares two individuals and selects one, which has a better cost, for the next generation. A crossover operator is the most "genetic" algorithm. A simple crossover operator, single-point crossover, selects two

individuals, who have survived into the next generation, cuts each individual string at the same point and exchanges a part of each string under crossover probability p_c. A mutation operator flips a digit in a string under mutation probability p_m. The mutation applies to all digits in all individual strings after operations of reproduction and crossover. The simple GA operation is convergent when the best individual through a population is not improved even if the generation proceeds.

As an example, a flight-path optimization problem may be expressed as

$$V = \max_u J(x(t_f), t_f) \tag{8}$$

with system governing Eq. 3, initial conditions

$$\chi(x(t_0), t_0) = 0 \tag{9}$$

and terminal constraints

$$\psi(x(t_f), t_f) = 0 \tag{10}$$

In a simple GA operation, a cost function is defined as:

$$J_{fit} = J(x(t_f), t_f) - k\psi(x(t_f), t_f)^T \psi(x(t_f), t_f) \tag{11}$$

that is, the terminal constraint (10) is dealt with using penalty terms in (11) with weighting coefficients, k. The dynamic system (3) can be numerically integrated to obtain the state variables at the terminal time. Some of the initial states required are found using (9); remaining initial states are optimized parameters. Other parameters optimized in the simple GA operation are the final time (if time is free) and discretized control variables.

Using GA for the solution of the outer-loop problem of determining the optimal sequence of events for the mission, q^*, seems natural since every event in the set of categorical variables Q can be represented by an integer and hence any sequence q can be represented by a string of 0's and 1's. Alternatively, the matrix Δ of Eqs. (5) and (6), which represents the solution to the integer programming problem, i.e. it determines the optimal sequence of events q^* through (5), is composed entirely of elements which are either 0 or 1. It could be encoded into a string for use in a GA. We have experience with the first approach, which will be described in the next section.

The advantages of using GA for numerical optimization are well-known: (i) GA's are generally fast and robust; (ii) it is generally straightforward to create a solver using GA, e.g. in comparison to deriving analytical necessary conditions using the calculus of variations, and (iii) GA's need no a priori information about the solution, in fact they begin from a randomly generated guess.

3.1 Solution of constrained problems, i.e. those with unallowed transitions

In order to use a GA as an outer-loop problem solver the GA must accommodate information regarding unallowed transitions, i.e. transitions from event q_i to event q_k such that the switching function $S(q_i, q_k) = 0$, which is equivalent to having element A_{ik} of the adjacency matrix $\mathbf{A}$ equal to 0. One of the very attractive characteristics of the GA method of optimization is that the initial population is created randomly, that is, no initial information regarding the optimal solution is required by the method. However, this will guarantee that the initial population will include unallowed strings, those with unallowed transitions of events. It is also certain that the GA processes of

crossover and mutation will *create* unallowed strings from parent strings that are free of them. The question for future research is how best to remove unallowed strings from the population. We believe there are two possible satisfactory approaches. The first approach is the obvious one; filter out unallowed strings as soon as they are created. This approach may be problematic as it creates the need for immediate replacement of lost strings if the population size is not to decrease. An alternative that we have derived is to use "fitness-augmentation" to remove unallowed strings via natural selection. That is, we give unallowed strings a supplemental cost so that their total objective function is so large that they are unlikely to propagate into the next generation.

4 Example: multi-asteroid interception mission

A spacecraft is to depart Earth orbit about the Sun and visit 3 asteroids out of a population of 8 asteroids in various orbits about the Sun. Impulsive velocity changes are allowed at Earth, at the first asteroid flyby (to arrange interception of the second asteroid) and at the second flyby (to arrange interception of the third asteroid). The objective is to minimize the sum of the three velocity changes ($\Delta V_1 + \Delta V_2 + \Delta V_3$). The times of interception of the three asteroids are free variables to be chosen by the optimizer. No asteroid may be visited more than once.

To simplify the problem the orbit of the Earth is assumed circular and the asteroids are in circular orbits in the ecliptic plane, making the problem two-dimensional. Canonical units are used in which 1 AU is one distance unit (DU) and 2π time units (TU) equal the period of an orbit at 1 AU (i.e. 1 year), thus the gravitational parameter $\mu_{\text{Sun}} = 1\,\text{DU}^3/\text{TU}^2$. The system equations of motion for the spacecraft then become:

$$\begin{aligned} \frac{\mathrm{d}r}{\mathrm{d}t} &= v_r \\ \frac{\mathrm{d}\theta}{\mathrm{d}t} &= v_\theta / r \\ \frac{\mathrm{d}v_r}{\mathrm{d}t} &= v_\theta^2 / r - 1/r^2 \\ \frac{\mathrm{d}v_\theta}{\mathrm{d}t} &= -v_\theta v_r / r \end{aligned} \tag{12}$$

If the times of interception of the three asteroids are t_1, t_2, t_3, respectively, then the interception conditions are:

$$r(t_i) = r_{\text{ast}}(t_i), \theta(t_i) = \theta_{\text{ast}}(t_i), i = 1, 2, \text{ or } 3. \tag{13}$$

At the initial time, the polar coordinates of the spacecraft (and Earth) are $r = 1$ AU and $\theta = 0$. The initial locations of the eight asteroids are given in Table 1.

Table 1 Initial positions of asteroids

Asteroid	1	2	3	4	5	6	7	8
r	1.450	1.699	2.001	1.490	1.650	1.730	1.960	1.700
θ	0.377	0.564	0.761	0.430	0.512	0.617	0.708	0.812

4.1 Solution via GA + NLP

The outer-loop problem is to determine the choice of asteroids to be visited and the order of interception. This is a particularly straightforward problem to convert for solution by GA; the chromosome for the problem is a string of 9 binary numbers, for example |001| 110 |010|, representing a mission to asteroids 2, 7 and 3 (since the asteroids are labeled 1–8). The population size was arbitrarily chosen to be 30. The initial population is randomly generated. The inner-loop problem is to determine the optimal trajectories, for the three "legs" of the mission; Earth to first asteroid, first to second asteroid, and second to third asteroid, satisfying conditions (13) while minimizing total ΔV. The R–K parallel shooting method described in Sect. 2.1 is used. We chose to use 35 segments for each of the three legs of the flight; thus there are 108 problem nodes. With 4 state and no control variables in the equations of motion (12) there are 432 states as NLP parameters. There are additionally three impulses applied and for each impulse the magnitude, direction and time of the impulse (or asteroid encounter) need to be specified, yielding 9 more variables. The NLP problem thus consists of optimally determining $432 + 9 = 441$ NLP parameters subject to $105 \times 4 = 420$ nonlinear constraint equations, 6 interior-point constraints (13) representing the interceptions, 4 constraints describing the continuity of the two state (position) variables before and after the impulses applied at the 1st and 2nd asteroids, 4 constraints relating the discontinuity of the state (velocity) variables to the impulses applied at the 1st and 2nd asteroids, and various initial conditions.

The progress in time of the solution algorithm is shown in Table 2. The first row (generation 1) results differ from subsequent results because the first generation consists of randomly generated strings. It indicates that of the 30 asteroid visit sequences generated 11 are infeasible, which here implies that these 11 strings do not specify

Table 2 Progress of the GA solution

Generation	Infeasible sequences	Cumulative evaluations	Best cost (DV) DU/TU	Average cost (DV) DU/TU	Best sequence found
1	11	19	0.219	20.1805	8-1-4
2	6	36	0.216	15.2037	5-1-8
3	12	44	0.1876	21.828	8-5-4
4	9	57	0.1876	16.8529	8-5-4
5	4	66	0.1876	6.9009	8-5-4
6	3	75	0.1672	5.2374	8-4-5
7	6	81	0.1672	11.8666	8-4-5
8	9	85	0.1672	18.4812	8-4-5
9	7	89	0.1672	20.1498	8-4-5
10	8	93	0.1672	18.478	8-4-5
11	8	95	0.1672	13.5151	8-4-5
12	2	98	0.1672	3.563	8-4-5
13	6	100	0.1672	11.8382	8-4-5
14	5	100	0.1672	10.1688	8-4-5
15	1	101	0.1672	1.869	8-4-5
16	0	101	0.1672	0.199	8-4-5
17	2	101	0.1672	3.5143	8-4-5
18	0	103	0.1672	0.1848	8-4-5
19	0	103	0.1672	0.1816	8-4-5
20	0	103	0.1672	0.1721	8-4-5

three different asteroids. For the 19 feasible strings optimal trajectories are found using the NLP based inner-loop problem solver described in Sect. 2.1. The best solution, visiting asteroids in the sequence 8-1-4, has a total ΔV of 0.219 DU/TU. The 30 individuals in the first generation are then subjected to the evolutionary processes of selection, crossover and mutation, as described in Sect. 3. The "fitness-augmentation" method described in Sect. 3.1 is used to remove unallowed strings via natural selection. This yields a new population of 30 individuals; 6 of which are infeasible. Only 17 new optimal trajectories need to be found by the inner-loop solver, as there is no need to determine solutions for either infeasible strings or new strings which are the same as strings found in a previous generation, bringing the cumulative number of inner-loop problem solutions at the 2nd generation to 36.

By the 6th generation we see in retrospect that the GA has found the optimal sequence of asteroids to visit (8-4-5) but the process is continued for a fixed term of 20 generations in the event a better sequence is located. The optimal trajectory is shown in Fig. 4. A total of 103 solutions of the inner-loop continuous optimal control problem are required. Note that a total enumeration of all of the possible ways in which 3 asteroids can be chosen, without repetition, from a set of 8 asteroids, would require 336 optimal trajectories. The GA + NLP hybrid optimal control problem solver thus locates the minimum without needing to find solutions for more than a fraction of the total possible solutions. (This fraction would be significantly smaller if the GA had been programmed to stop after the same best sequence had been found several times in a row, rather than proceeding for a full 20 generations.)

4.2 Solution via branch and bound + GA

The same problem has been solved using a different evolutionary algorithm. For the second solution the branch and bound (B&B) method (von Stryk and Glocker 2001) has been used as the outer-loop solver. B&B is an optimization method widely used in

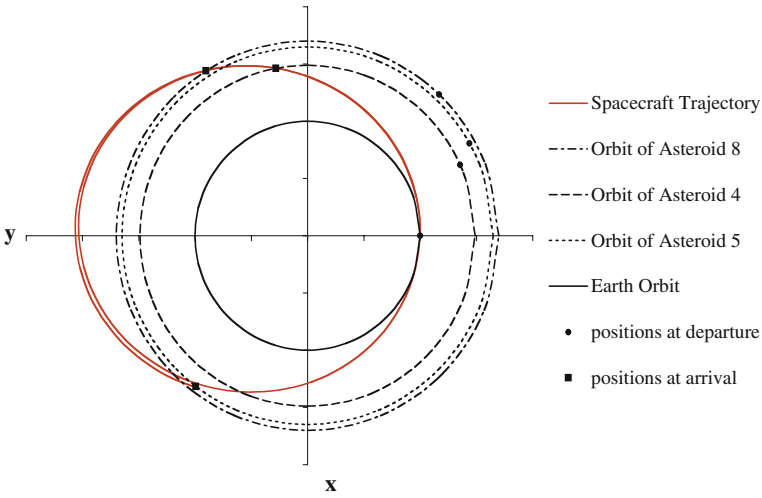

Fig. 4 Optimal asteroid interception trajectory for case 8-4-5

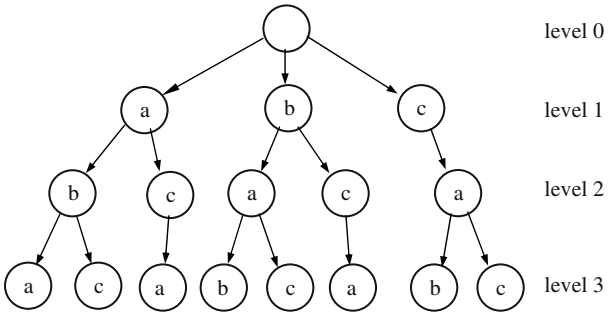

Fig. 5 Decision tree for branch and bound method

industrial applications such as process optimization and task scheduling. It has been applied recently for discrete optimization in the context of HOCP's.

The construction of an event sequence requires a decision process for each event that is placed in the sequence. The set of all the possible choices can be depicted graphically as a tree. Fig. 5 shows a decision tree with $N_{s1} = 3$ corresponding to the digraph shown in Figure 1. The notation for the states, q_i, is changed to i for clarity in the figure.

The tree traverse starts at level 0 by taking the root node as the current parent and selecting a child node from among those in level 1. The chosen node constitutes the first event in the sequence and becomes the parent for the next selection to be performed over its children in level 2. This process continues until the search arrives at one of the leaves at level N_{s1} when a complete sequence has been determined.

Every selection of a child node during the traverse yields a partial sequence that is a relaxed instance of a nominal problem, i.e. a complete sequence. Optimal control theory states that a problem with relaxed or fewer constraints has a smaller cost than the nominal problem. Hence, evaluating partial branches of the tree provides lower bounds that can be compared with the cost of a complete "incumbent" solution sequence. The incumbent solution is a feasible suboptimal sequence of length N_{s1} that can be easily found by intuition or experience. If the cost of any of the partial sequences is higher than that of the incumbent, the tree can be pruned at that node because no sequence proceeding from that point will have a smaller cost.

When the search arrives at a leaf and the complete sequence is found to be better than the incumbent, the newly found sequence becomes the new incumbent. This operation may indicate that performing a depth-first search on the tree may be more efficient than a breadth-first approach. Modeling is also a very important matter to ensure that relaxations of the problem exist in order to compute the cost of partial sequences.

An important feature of the method is that it is not heuristic. It finds the best solution while reducing the search space methodically. The performance of the method is affected primarily by the selection of the incumbent solution sequence. A poor incumbent will do little to prune the tree in a significant way causing the search to be similar to total enumeration. A good incumbent, however, will find the best global solution by performing a small number of candidate evaluations. A more subtle issue is the order of evaluation of nodes at a particular level. This can be done left-to-right

or vice versa. One search direction can be better than the other depending on the location of the leaf that completes the best sequence, which is not known a priori.

A GA was used as the inner-loop problem solver. The only GA parameters for this problem were the transfer times on the three legs of the trajectory. Given the transfer times the positions of the asteroids at the interception times can be calculated. With this information a Lambert's solution, including the magnitude of the impulse required to arrive at the asteroid in the given transfer time, can be found using the universal Lambert solver (Battin 1987). Because the solution is exact, the equations of motion (3) are not required. This is of course a great simplification in comparison to the integration of the equations of motion (3) that was required when the GA + NLP solver was used. However that solver, as previously mentioned, could easily accommodate more complicated systems, for example where low-thrust propulsion is used or where the attraction of a third body is included as a perturbation, while the approach using Lambert's method cannot be extended to such cases.

Two versions of relaxed problems were required by the B&B method. Fortunately for this case relaxed versions of the problem exist naturally and have a physical significance. The first relaxation was defined as the minimum-fuel intercept of only the first asteroid since any two-asteroid sequence that includes the first asteroid must have a greater or equal cost. Similarly, the second relaxed problem was defined as the minimum-fuel interception of the first two asteroids. Each GA was run for 100 generations. After 8 single-asteroid interception evaluations, 56 two-asteroid interception evaluations, and 132 three-asteroid interception evaluations the optimal sequence of asteroids to visit (8-4-5) was found with a cost $\Delta V = 0.1688$ AU/TU. The small discrepancy between the cost for the previous GA + NLP solution (0.1672 AU/TU) and that of the B&B + GA solution can be attributed to the way in which the GA solves the inner-loop problem. There is an imprecise resolution in the flight time parameters that necessarily results when the GA needs to describe a base 10 number with a binary number of reasonably modest length. That is, the GA found interception times for the 1st, 2nd, and 3rd asteroids of 2.92, 13.85 and 20.85 TU, respectively, to 2 decimal place accuracy. The nearly exact results obtained using the transcription of the problem into a NLP problem via R–K parallel shooting were 2.9114, 13.8469, and 20.8820 TU, respectively. Of course the inner-loop solver using transcription to a NLP problem is much more precise.

Note that even use of the B&B algorithm cannot guarantee location of the global minimum for MIOCP's with nonlinear dynamics subject to nonlinear constraints and with transitions at unspecified event times. Thus as an additional check, the B&B + GA solver was run for all 336 possible itineraries for visiting three asteroids out of the 8 available and this again confirmed the result that the 8-4-5 itinerary yields the global minimum.

5 Conclusions

The problem of optimal space mission planning has been described as a problem in hybrid optimal control theory. While no general methods exist for the solution of such problems two algorithms using evolutionary principles have been discussed and demonstrated in the solution of a simple but challenging orbit transfer problem. For hybrid optimal control problems perhaps the most relevant measure of performance is how many possible solutions need to be determined, in comparison to total enumeration

of the solution space, in order to locate the global minimum. By this measure both algorithms do quite well, so that these or similar algorithms have the potential to save mission planners much effort. Future work will apply the method to more challenging trajectories and cases where total enumeration is clearly impractical. The researchers' goal is to enable the evolutionary methods to create an optimal interplanetary trajectory autonomously from a catalog of possible maneuvers such as low-thrust arcs, impulses, and planetary flybys.

References

Battin, R.H.: An Introduction to the Mathematics and Methods of Astrodynamics. AIAA (1987)
Buss, M., Hardt, M., von Stryk, O.: Numerical solution of hybrid optimal control problems with applications in robotics. Proc. 15th IFAC World Congress on Automatic Control, Barcelona (2002)
Carroll, D.L.: Fortran Genetic Algorithm (GA) Driver (1998). http://www.staff.uiuc.edu/~carroll/ga.html
Enright, P.J.: Optimal finite thrust spacecraft trajectories using direct transcription and nonlinear programming. Ph.D. Thesis, University of Illinois at Urbana-Champaign (1991)
Enright, P.J., Conway, B.A.: Optimal finite-thrust spacecraft trajectories using collocation and nonlinear programming. J. Guidance, Control Dynamics **14**(5), 981–985 (1991)
Enright, P.J., Conway, B.A.: Discrete approximations to optimal trajectories using direct transcription and nonlinear programming. J. Guidance, Control, Dynamics **15**(4), 994–1002 (1992)
Gill, P. E., Murray, W., Saunders, M.A.: User's Guide for SNOPT 5.3: A Fortran Package for Large-scale Nonlinear Programming. Stanford Univ., Stanford, CA (Jan. 1998)
Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA, Addison-Wesley (1989)
Herman, A.L., Conway, B.A.: Direct optimization using collocation based on high-order Gauss-Lobatto Quadrature Rules. J. Guidance, Control, Dynamics **19**(3), 592–599 (1996)
Ross, I.M., Fahroo, F.: Pseudospectral knotting methods for solving optimal control problems. J. Guidance, Control, Dynamics **27**(3) 397–405 (2004)
Ross, I.M., D'Souza, C.N.: A hybrid optimal control framework for mission planning. J. Guidance, Control, Dynamics **28**(4), 686–697 (2005)
von Stryk, O., Glocker, M.: Decomposition of mixed-integer optimal control problems using branch and bound and sparse direct collocation. In: Engell, S., Zaytoon, J. (eds.) Proc. 4th Int. Conf. on Automation of Mixed Processes: Hybrid Dynamic Systems, Dortmund, pp. 99–104 (2000)
von Stryk, O., Glocker, M.: Numerical mixed-integer optimal control and motorized traveling salesman problems. Eur. J. Control **35**(4), 519–533 (2001)