MIDACO on MINLP Space Applications

Martin Schlueter
Division of Large Scale Computing Systems, Information Initiative Center,
Hokkaido University, Sapporo 060-0811, Japan
{info@midaco-solver.com}

Sven O. Erb
European Space Agency (ESA), ESTEC (TEC-ECM),
Keplerlaan 1, 2201 AZ, Noordwijk, The Netherlands
{sven.erb@esa.int}

Matthias Gerdts
Institut fuer Mathematik und Rechneranwendung, Universität der Bundeswehr,
München, D-85577 Neubiberg/München, Germany
{matthias.gerdts@unibw.de}

Stephen Kemble
Astrium Limited, Gunnels Wood Road, Stevenage, Hertfordshire, SG1 2AS, United Kingdom
{Stephen.KEMBLE@astrium.eads.net}

Jan-J. Rückmann
School of Mathematics, University of Birmingham, Birmingham B15 2TT, United Kingdom
{j.ruckmann@bham.ac.uk}

November 15, 2012

Abstract

A numerical study on two challenging MINLP space applications and their optimization with MIDACO, which is a recently developed general purpose optimization software, is presented. The applications are in particular the optimal control of the ascent of a multiple-stage space launch vehicle and the space mission trajectory design from Earth to Jupiter using multiple gravity assists. Additionally an NLP aerospace application, the optimal control of an F8 aircraft manoeuvre, is furthermore discussed and solved. In order to enhance the optimization performance of MIDACO a hybridization technique, coupling MIDACO with a SQP algorithm, is presented for two of the three applications. The numerical results show, that the applications can be solved to their best known solution (or even new best solutions) in a reasonable time by the here considered approach. As the concept of MINLP is still a novelty in the field of (aero)space engineering, the here demonstrated capabilities are seen as promising.

Keywords: MIDACO, MINLP, NLP, SQP, Hybrid Optimization, Space Application.

1 Introduction

In this contribution the optimization of space applications via a mixed integer nonlinear programming (MINLP) approach is presented. A mathematical formulation of a general MINLP is given in (1), where $f(x, y)$ denotes the objective function to be minimized. In (1), the equality constraints are given by $g_{1,...,me}(x, y)$ and the inequality constraints are given by $g_{me+1,...,m}(x, y)$. The vector x contains the continuous decision variables and the vector y contains the discrete decision variables. Furthermore, some box constraints as x_l, y_l (lower bounds) and x_u, y_u (upper bounds) for the decision variables x and y are considered in (1).

$$
\begin{aligned}
&\text{Minimize} \quad f(x,y) && (x \in \mathbb{R}^{n_{con}},\ y \in \mathbb{Z}^{n_{int}},\ n_{con},\ n_{int} \in \mathbb{N}) \\
&\text{subject to:} \quad g_i(x,y) = 0, && i = 1,...,m_e \in \mathbb{N} \\
&\qquad\qquad\ \ g_i(x,y) \geq 0, && i = m_e+1,...,m \in \mathbb{N} \\
&\qquad\qquad\ \ x_l \leq x \leq x_u && (x_l,\ x_u \in \mathbb{R}^{n_{con}}) \\
&\qquad\qquad\ \ y_l \leq y \leq y_u && (y_l,\ y_u \in \mathbb{N}^{n_{int}})
\end{aligned}
\tag{1}
$$

Considering both, continuous and integer optimization variables, and without any assumptions (like convexity or differentiability) on the objective function or constraints, MINLP problems are one of the most general and most difficult types of optimization problems. Due to this generality, a wide range of applications can be modelled as MINLP with an intriguing additional potential in contrast to purely continuous or discrete formulations. While MINLP research is well established in academic areas like chemical engineering (see Kocis and Grossmann [5]) and gaining more and more attention in other areas (e.g. Bio-Informatics, see Maria et. al. [6]), its adaption to space and aerospace applications remain very small so far to the best knowledge of the authors.

This papers presents a numerical study on three challenging (aero)space applications that are solved by the global optimization software MIDACO, which was recently developed for general MINLP problems. Taking advantage of the MINLP capabilities of MIDACO, the two space applications are formulated as MINLP. Those space applications are in particular the optimal control of a multiple-stage launch vehicle (based on a Delta III rocket by *The Boing Company*) and the design of an interplanetary space mission trajectory from Earth to Jupiter (based on the Galileo mission from 1989 by *NASA*) and. In case of the space mission application planetary multi-gravity assist manoeuvrers are considered, whereas the flyby planet candidates imply discrete decision variables to be optimized. In case of the launch vehicle application the model is extended by some discrete decision variables, which express different booster configuration scenarios. Additionally to the MINLP space applications the optimal control of an F8 aircraft [8] manoeuvre (formulated as NLP) is presented here. In order to further enhance the performance of the stochastic MIDACO algorithm, a hybridization approach of MIDACO with a deterministic local SQP algorithm is proposed in three out of the two applications.

Note that furthermore to the mathematical model descriptions given here, the numerical implementations of the proposed applications (not including the MIDACO and SQP software) can be freely downloaded at `http://midaco-solver.com/applications.html`.

This paper is structured as follows: Firstly, a brief introduction on the MIDACO software and its hybridization with a SQP algorithm is given in Section 2.1. Secondly, numerical results by MIDACO on the above mentioned space and aerospace applications are presented and compared to available reference solutions in Section 3, Section 4 and Section 5. Finally, some conclusions are drawn.

2 MIDACO Software

This section provides a brief introduction to MIDACO, which stands for *Mixed Integer Distributed Ant Colony Optimization*. The MIDACO software implements a global optimization algorithm for black-box MINLP problems based on an extended ant colony optimization algorithm (see [10] and [11]) coupled with the oracle penalty method (see [12]) for constraint handling. The software is available in several programming languages (in esp. Fortran, C/C++ and Matlab) and comprehensive information as well as a free test version are available at the MIDACO homepage [9]. A recent numerical study of MIDACO on 100 MINLP benchmark problems (see [13]) reveals the strength of the software compared to other established MINLP solvers.

Here the relevant algorithmic parameters for MIDACO, that appear in this contribution are listed together with a brief description.

Seed - Initial seed for internal pseudo random number generator within MIDACO. The *Seed* determines the sequence of pseudo random numbers sampled by the generator. Therefore MIDACO runs using an identical *Seed*, will produce exactly the same results (executed on the same computer under identical compiler conditions). As the *Seed* may be an arbitrary integer greater or equal to zero, the user can easily generate (stochastically) different runs, using a different *Seed* parameter. The advantage of a user specified random seed is, that promising runs can easily be reproduced by knowing the applied *Seed* parameter. This is in esp. useful, if a run must be stopped out of some reason and should be restarted again.

Qstart - This parameter allows the user to specify the quality of the starting point. If *Qstart* is set greater than 0, the initial population of iterates (also called ants) is sampled closely around the starting point. In particular, the standard deviation for continuous variables is set to $|x_l - x_u|/Qstart$ and the mean is set to the corresponding dimension of the starting point. For integer variables the standard deviation is set to $max\{|y_l - y_u|/Qstart,\ 1/\sqrt{Qstart}\}$ to avoid a too tight sampling. The greater *Qstart* is selected, the more closely does MIDACO search around the starting point. This option is very useful to refine previously calculated solutions. It is important to note, that this option does not shrink the search space. The original bounds x_l, y_l and x_u, y_u are still valid, only the initial population of ants is specifically focused within these bounds.

Autostop - This parameter activates an internal stopping criteria for MIDACO. While it is recommended, that the user will run MIDACO for a fixed time or evaluation budget, this option allows the software to stop the optimization run by itself. *Autostop* defines the amount of internal restarts in sequence, that did not reveal an improvement in the objective function value. The greater *Autostop* is selected, the higher the chance of global optimality, but also the longer the optimization run. As *Autostop* can be selected any integer greater or equal to zero, it gives the user the freedom to compromise between global optimality and cpu run time to his or her specific needs.

Oracle - This parameter specifies a user given oracle parameter Ω [12] to the penalty function within MIDACO. If *Oracle* is selected not equal to zero, MIDACO will use the *Oracle* as long as a better feasible solution has been found. This option can be useful for problems with difficult constraints where some background knowledge on the problem exists.

2.1 Hybridization with SQP

In Section 3 and Section 5 MIDACO has been used in a coupled approach together with an SQP algorithm for NLP (in esp. SQP-Filtertoolbox available from Prof. Gerdts, http://www.unibw.de/lrt1/gerdts/software). The most straight forward approach was applied for the hybridization, which is the splitting of the optimization process into two separate phases: First MIDACO is applied on the original MINLP problem using the lower bounds $(x_{lower},\ y_{lower})$ as starting point, second SQP is applied on the MINLP problem with fixed integer variables using the best solution revealed by MIDACO as starting point. The mathematical formulation of the MINLP problem (1) with fixed integer variables is given in 2.

$$
\begin{aligned}
&\text{Minimize} \quad f^{y}(x) \qquad\qquad (x \in \mathbb{R}^{n_{con}}),\\
&\\
&\text{subject to:} \quad g_i^{y}(x) \;=\; 0, \quad (i \;=\; 1,...,m_e),\\
&\qquad\qquad\quad\; g_j^{y}(x) \;\geq\; 0, \quad (j \;=\; m_e+1,...,m),\\
&\qquad\qquad\qquad x_l \leq x \leq x_u, \quad (x_l, x_u \in \mathbb{R}^{n_{con}} : x_l \leq x_u),\ y \in \mathbb{Z}^{n_{int}}.
\end{aligned}
\tag{2}
$$

Note that in Equation 2 the integer decision variables y from MINLP (1) are considered as parameters and not as optimization variables. The concrete values for y are given by the solution of the MINLP (1) revealed by MIDACO, while the SQP algorithm only optimizes the continuous variables x.

3 Multiple-Stage Launch Vehicle Ascent Problem

The ascent of a multiple-stage launch vehicle is considered here. The model is based on a Delta III rocket (*The Boeing Company*) and was originally introduced by Benson [1]. In its original (continuous) formulation, the model is used as a benchmark in the open literature Huntington [3] and in well known optimal control software packages like GPOPS [7].

Here, the original model as given in GPOPS [7] is extended by additional mixed integer decision variables and nonlinear constraints, creating a challenging mixed integer multi stage optimal control problem. While in the original formulation the type and number of strap on boosters used for the rocket propulsion is fixed, those two engineering design aspects are formulated as discrete decision variables here. It can be shown, that by introducing those additional degrees of freedom, significant improvements in the objective function can be gained in comparison to the original model. Additionally incorporated constraints on the maximal dynamic pressure for the vehicle and a virtual financial budget (based on the type of strap on boosters employed) ensure, that the feasible solutions are still reasonable.

The objective in this application is to maximize the remaining fuel of the vehicle while maneuvering it from the ground to a low earth target orbit. In the following, the model formulation in GPOPS [7] is closely followed while the above mentioned extensions are especially highlighted in an individual subsection.

Note that the implementation of this application discussed in the following can be downloaded at http://midaco-solver.com/applications.html.

3.1 Vehicle Properties

The launch vehicle consists of two main stages and contains nine strap-on solid rocket boosters. The flight of the vehicle to its target orbit can be divided into four different phases. The first flight phase considers the vehicle on the ground at time t_0. The main engine and a number of boosters

ignite (the concrete number of boosters is considered an optimization variable here). At time t_1 the number of boosters ignited at t_0 are depleted and the remaining dry mass is jettisoned. In the following second flight phase, the remaining strap-on boosters ignite and are depleted at time t_2. The third flight phase does only consider propulsion by the main engine of the vehicle stage 1. The fourth flight phase begins when the main engine fuel is finished and the dry mass associated with the main engine is ejected at time t_3. During flight phase four only the main engine of the vehicle stage 2 is used for propulsion. The flight phase four ends at time t_4, when the vehicle reaches the desired low earth target orbit. Note that the solid boosters and main engine burn for their entire duration (meaning t_1, t_2, and t_3 are fixed), while the second stage engine is shut off when the target orbit is achieved, therefore t_4 is an optimization variable. The mass and propulsion properties of the two vehicle stages are taken from GPOPS [7] and listed in Table 1.

Table 1: Vehicle mass and propulsion properties

	Stage 1	Stage 2
Total Mass (kg)	104380	19300
Propellant Mass (kg)	95550	16820
Engine Thrust (N)	1083100	110094
Specific Impulse (sec)	301.7	467.2
Number of Engines	1	1
Burn Time (sec)	7261	700

Dynamic Model

The equations given in 3 express the Cartesian coordinates (earth-centered) of a non-lifting mass point in flight over a spherical rotating planet

$$\begin{aligned} \dot{r} &= v, \\ \dot{v} &= -\frac{\mu}{\|r\|^3} + \frac{T}{m}u + \frac{D}{m}, \\ \dot{m} &= -\frac{T}{g_0 I_{sp}}, \end{aligned} \tag{3}$$

where $r = (x, y, z)'$ is the (earth-centered) Cartesian position of the mass point, $r = (v_x, v_y, v_z)'$ is the velocity, μ is the gravitational parameter, T is the vacuum thrust, m is the mass, g_0 is the acceleration due to gravity at sea level, I_{sp} the specific impulse of the engine, $u = (u_x, u_y, u_z)'$ is the thrust direction and $D = (D_x, D_y, D_z)'$ is the drag force. The drag force is defined as

$$D = -\frac{1}{2}\rho S C_D \|v_{rel}\| v_{rel} \ , \tag{4}$$

where C_D is the drag coefficient, S is the reference area, ρ is the atmospheric density and v_{rel} is the earth relative velocity, where v_{rel} is given as

$$v_{rel} = v - \Omega \times r, \tag{5}$$

where Ω is the angular velocity of the earth relative to the inertial reference frame. The atmospheric density is modelled as the exponential function

$$\rho = \rho_0 e^{\frac{-h}{H}}, \tag{6}$$

where ρ_0 is the atmospheric density at sea level, $h = \|r\| - R_e$ is the altitude, R_e is the equatorial radius of the earth and H is the density scale height. Table 2 contains the numerical values for the constants used in the vehicle model.

Table 2: Constants used in the launch vehicle model

Constant	Value
Payload mass (kg)	4164
S (m^2)	$4\ \pi$
C_D	0.5
ρ_0 (kg/m^3)	1.225
H (m)	7200
t_1 (sec)	75.2
t_2 (sec)	150.4
t_3 (sec)	261
R_e (m)	6378145
Ω (rad/s)	$7.29211585 \times 10^{-5}$

The launch vehicle starts on the ground at rest (relative to the earth) at time t_0, so that the earth centered initial conditions are

$$\begin{aligned} r(t_0) &:= r_0 = (R_e cos\Phi_0, 0, R_e sin\Phi_0)', \\ v(t_0) &:= v_0 = \Omega \times r_0, \\ m(t_0) &:= m_0 = 301454(kg), \end{aligned} \tag{7}$$

where $\Phi_0 = 28.5^\circ$ and corresponds to the geocentric latitude of Cape Canaveral (Florida) and it is arbitrarily assumed that the inertially fixed axes are such that the initial inertial longitude is zero. The terminal constraints define the target geosynchronous transfer orbit, which is defined in terms of orbital elements as

$$\begin{aligned} a_f &= 24361.14 \text{ km}, \\ e_f &= 0.7308, \\ i_f &= 28.5^\circ, \\ \Omega_f &= 269.8^\circ, \\ \omega_f &= 130.5^\circ, \end{aligned}$$

where a is the semimajor axis, e is the eccentricity, i is the inclination, Ω is the inertial longitude of the ascending node and ω is the argument of perigee. The true anomaly v is considered free, as no location in the terminal orbit is specified as constraint. Besides the primary constraint of reaching the terminal orbit, a state path constraint keeps the altitude of the vehicle above the earth surface and is given as

$$|r| \geq R_e, \tag{8}$$

where R_e is the equatorial radius of the earth. In contrast to the original model formulation in GPOPS [7], no equality path constraint of the form

$$\|u\|^2 = u_x^2 + u_y^2 + u_z^2 = 1 \tag{9}$$

is necessary here, because the Pitch and Yaw system is applied to control the vehicle that inherently satisfies the above equality path constraint. The transformation from Pitch (Ψ) and Yaw (Φ) to the cartesian thrust direction $u = (u_x, u_y, u_z)'$ is given by

$$\begin{aligned} u_x &= cos(\Psi)cos(\Theta), \\ u_y &= cos(\Psi)sin(\Theta), \\ u_z &= sin(\Psi), \end{aligned} \tag{10}$$

where $\Psi \in [-\pi, \pi]$ and $\Theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. The model further contains linkage constraints between the different phases regarding the position r, the velocity v and the mass m of the vehicle. Those linkage constraints are active at the end of phases 1,2 and 3 and the start of phases 2, 3 and 4,

respectively as

$$\begin{aligned} r^{(p)}(t_f) - r^{(p+1)}(t_0) &= 0, \\ v^{(p)}(t_f) - v^{(p+1)}(t_0) &= 0, \\ m^{(p)}(t_f) - m^{(p)}_{dry} - m^{(p+1)}(t_0) &= 0, \end{aligned} \tag{11}$$

where the subscript (p) denotes the phase number (in esp. $p = \{1, 2, 3\}$) and t_0 and t_f denote the start and final time points of the corresponding phase. The linkage constraints force the position r and velocity v to be continuous regarding the phase transition. The linkage constraint on the mass $m^{(p)}$ at each of the phase interfaces corresponds with an instantaneous drop of the dry mass $m^{(p)}_{dry}$ of the particular phase (p). Therefore the mass trajectory is not continuous at the stage interfaces, when mass is ejected. Here, mass drops at the ends of phases 1,2 and 3, when the dry mass of the strap-on boosters or the first main stage is depicted. In contrast to the original formulation of the model in GPOPS [7], the amount of dry mass associated with the strap-on boosters dropped at the ends of phases 1 and 2 depends on the number of boosters chosen, which is an integer decision variable (see Subsection 3.2).

The objective of the problem is to find an optimal control (and corresponding trajectory) that maximizes the remaining mass of the vehicle at the end of phase 4. This objective is expressed as minimization of the cost functional J given as

$$J = -m^{(4)}(t_f), \tag{12}$$

subject to the above defined conditions and constraints.

3.2 Mixed Integer Extensions

The launch vehicle model is extended by some discrete decision variables regarding the number and type of strap-on boosters used. While in the original formulation in GPOPS [7] the number of strap-on boosters is considered fixed (6 booster for phase 1 and 3 booster for phase 2), here the number of strap-on booster applied in the first phase is a decision variable and denoted by $B_1 \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The number B_2 of strap-on booster applied in the second phase is then recursively computed by $B_2 = 9 - B_1$ and therefore not considered a decision variable. Further to the number of boosters applied, also the type of booster is considered to be variable here. For simplicity and practical relevance the selection of booster types is restricted to phase 1 and 2. Hence, all boosters of phase 1 or 2 are assumed to be of the same type. Based on the type of strap-on booster defined in the original formulation, four new types are generated. Those four new types do vary in 10 and 25 percentage respectively to the properties of the original booster defined in GPOPS [7]. Table 3 lists all five possible strap-on booster types with their properties. Note, that in Table 3 the booster type 3 corresponds to the original booster considered in GPOPS [7].

Table 3: Abbreviations for Table

Type	Thrust Power (N)	Mass $_{Total}$ (kg)	Mass $_{Propellant}$ (kg)	Cost
1	471375	14468	12758	0.75
2	565650	17361	15309	0.90
3	628500	19290	17010	1.00
4	691350	21219	18711	1.10
5	785625	24113	21263	1.25

Table 3 also introduces a new property for the five considered booster types. This is a virtual cost which is based on the original strap-on booster. Hence the cost for booster type 3 is defined as 1.00 while the costs for all other boosters correspond with their 10 or 25 percentage property in- or decrease. The here introduced cost property of the booster types is later used to define some financial constraint that restricts the feasible choice of boosters.

3.3 Additional Constraints

Additional constraints are considered here. This is a maximal dynamic pressure constraint which ensures, that the dynamic pressure on the launch vehicle never exceeds 50.000 N/m^2. This constraint is to ensure that the choice of boosters implies a reasonable dynamic pressure behavior that does not damage the space vehicle. The dynamic pressure constraint is given by

$$\frac{1}{2}P||v||^2 \leq 50000, \tag{13}$$

where P is the atmospheric pressure calculated as

$$P = 1.249512\ e^{-\frac{||r||}{6900}}. \tag{14}$$

A further constraint is introduced as maximal financial budget regarding the type of strap-on boosters. The maximal available budget is considered here as 9, which is based on the original model formulation applying 9 boosters of type 3 (which has cost 1 as listed in Table 3). The financial budget constraint is formulated as

$$B_1T_1 + B_2T_2 \leq 9, \tag{15}$$

where $B_2 = 9 - B_1$ is the number of strap-on boosters used in the second phase. Last, a vertical take-off constraint is imposed which ensures, that the vehicle launches vertical during the first five seconds of its flight. This constraint is considered a security procedure that increases the realistic relevance of the launch vehicle start. The vertical take-off constraint is imposed as a fixed control $\bar{u} = (\bar{u}_x, \bar{u}_y, \bar{u}_z)'$ during the first five seconds of phase 1, where $\bar{u}$ is given by

$$\bar{u} = \frac{r}{||r||}. \tag{16}$$

3.4 Numerical Results

To solve the above described (mixed integer multi stage) optimal control problem, an reduced direct approach is followed, where a Runge Kutta method of order 2 is applied to integrate the ordinary differential equations 3. A discretization of 15 grid points is applied in each of the four flight phases, which is identical to the number of grid points used in GPOPS [7]. The resulting MINLP has then 128 decision variables, where 3 of them are discrete (this is B_1, T_1, T_2; see Table 4), and 127 constraints, where 5 of them are equality constraints defining the target orbit. The integer complexity of the MINLP is 250 ($10 \times 5 \times 5$).

A hybrid optimization strategy is implemented. This strategy first applies MIDACO to search the MINLP formulation on the full mixed integer search domain for a fixed time budget. In a second step, an SQP algorithm (in esp. SQP-Filtertoolbox available from Prof. Gerdts, `http://www.unibw.de/lrt1/gerdts/software`) is applied, using the solution given by MIDACO as initial starting point. For the SQP method the 3 discrete decision variables are fixed to the values given by the MIDACO solution, hence the SQP is applied only on the continuous search domain of the problem and considered a refinement technique for the MIDACO solution. More details on the hybridization of MIDACO and SQP can be found in Section 2.1.

Before optimizing the full MINLP an investigation on the impact of the discrete decision variables is performed. Table 5 presents the best known results for the launch vehicle problem in respect to different combinations of discrete variables that have been fixed. Only feasible combinations with $B_1 \geq 6$ are considered in Table 5. For integer combinations with $B_1 = 9$ results are only reported in respect to T_1 because the influence of T_2 is literally zero in this scenario ($B_2 = 9 - B_1$). Table 4 contains the abbreviations for Table 5. Note, that in Table 5 the integer combination $\{B_1 = 6, T_1 = 3, T_2 = 3\}$ expresses the original booster configuration as assumed in GPOPS [7]. While in GPOPS [7] an optimal objective function of 7529.71kg is reported, here a value of 7504.48kg is given to the corresponding integer combination $\{6, 3, 3\}$. This difference find its explanation in the additional constraints and different integration approach applied here.

From the results of Table 5 it can be seen, that the discrete booster decision variables do have a significant impact on the objective function. In esp. seven different combinations can be identified, that improve the result in respect to the original combination of $\{6, 3, 3\}$. The best known result in Table 5 corresponds to the integer combination of $\{8, 3, 3\}$. The non-intuitive and diverse impact of the different integer combination on the solution exemplifies the complexity of this MINLP formulation very well.

Table 4: Abbreviations for Table 5

Abbreviation	Explanation
Booster-Config.	Booster Configuration: B_1, T_1, T_2
B_1	Number of active booster in first phase
T_1	Type of booster used in first phase
T_2	Type of booster used in second phase
Best known $f(x, y)$	Best known objective function for corresponding booster configuration

Table 5: Enumeration over all (feasible) booster configurations with $B_1 \geq 6$

Booster-Config.				Booster-Config.			
B_1	T_1	T_2	Best known $f(x, y)$	B_1	T_1	T_2	Best known $f(x, y)$
6	1	1	-6685.71	8	1	1	-6848.21
6	1	2	-6808.53	8	1	2	-6900.99
6	1	3	-6884.45	8	1	3	-6935.36
6	1	4	-6955.92	8	1	4	-6969.11
6	1	5	-7055.32	8	1	5	-7018.56
6	2	1	-7075.93	8	2	1	-7297.53
6	2	2	-7195.10	8	2	2	-7228.32
6	2	3	-7269.14	8	2	3	-7381.77
6	2	4	-7339.15	8	2	4	-7414.42
6	3	1	-7315.13	8	2	5	-7321.22
6	3	2	-7431.81	8	3	1	**-7565.08**
6	*3*	*3*	*-7504.48*	8	3	2	**-7614.97**
6	4	1	**-7539.17**	8	3	3	**-7647.50**
7	1	1	-6789.90	9	1	-	-6855.30
7	1	2	-6883.25	9	2	-	-7324.23
7	1	3	-6942.60	9	3	-	**-7599.88**
7	1	4	-6999.74				
7	1	5	-7081.49				
7	2	1	-7213.85				
7	2	2	-7303.58				
7	2	3	-7360.66				
7	2	4	-7415.64				
7	2	5	-7494.50				
7	3	1	-7271.36				
7	3	2	**-7556.82**				
7	3	3	**-7612.70**				

Table 6 and Table 6 present the results of 30 individual test runs on the multiple-stage launch vehicle problem using a fixed budget of 600 (10 Minutes) respectively 7200 (2 Hours) seconds for MIDACO. For every run, a different random seed is used for MIDACO. The different integer combinations and objective function values to the solutions found by MIDACO are reported along the number of function evaluation and cpu-time needed by MIDACO. The solutions by the SQP method, which uses the MIDACO solutions as starting points, are reported together with the number of function evaluation (including those for gradient approximations) and cpu times. The

SQP algorithm were called with a maximum number of 1000 iterations. Note that the SQP may stop earlier, if its convergence criteria is satisfied, while MIDACO always performs over its defined cpu-time budget.

Table 6: 30 runs by MIDACO (max time = 600) + SQP (max iter=1000)

	Booster-Config.			SQP			MIDACO		
Run	B_1	T_1	T_2	$f(x,y)$	Eval	Time	$f(x,y)$	Eval	Time
1	8	3	3	-7647.50	327060	322.6	-6857.99	222866	600.0
2	9	3	1	-7599.88	274208	292.9	-7163.86	239657	600.0
3	9	3	5	-7599.88	280700	376.8	-7065.85	235807	600.0
4	9	3	5	-7599.88	271308	259.6	-6972.28	198351	600.0
5	8	3	3	-7647.50	306974	282.5	-6963.09	281567	600.0
6	8	3	3	-7647.50	270452	258.3	-6920.35	269536	600.0
7	9	3	5	-7599.88	269558	262.1	-7038.81	262407	600.0
8	8	3	3	-7647.50	281106	264.3	-6820.98	270953	600.0
9	7	3	3	-7612.85	350416	373.0	-6956.02	261879	600.0
10	8	3	3	-7647.50	145076	139.4	-6996.72	268385	600.0
11	8	2	5	-7462.25	330614	314.2	-6918.20	266153	600.0
12	8	3	3	-7647.50	352408	336.5	-6889.87	266954	600.0
13	9	3	5	-7599.88	297954	377.8	-6972.39	258888	600.0
14	8	3	3	-7647.50	339916	368.0	-6818.03	208452	600.0
15	9	3	2	-7599.88	350742	321.5	-7024.97	272402	600.0
16	8	3	3	-7647.47	359535	334.2	-6914.47	270246	600.0
17	7	3	3	-7612.85	362934	367.2	-6977.27	239453	600.0
18	9	3	1	-7599.88	310868	315.7	-6954.59	258475	600.0
19	9	3	5	-7599.87	363478	369.6	-7036.21	245399	600.0
20	9	3	2	-7599.88	343712	351.9	-6797.39	257383	600.0
21	8	3	3	-7647.50	334478	337.0	-6971.25	252131	600.0
22	8	3	3	-7647.50	306314	311.1	-6852.83	254798	600.0
23	8	3	3	-7647.50	305118	299.1	-6929.07	250874	600.0
24	9	3	3	-7599.88	280658	281.6	-7031.31	254307	600.0
25	8	3	3	-7647.50	270780	252.8	-6911.24	212809	600.0
26	6	4	1	-7539.17	364376	335.3	-6814.24	267337	600.0
27	9	3	5	-7599.88	323402	297.5	-6834.61	273553	600.0
28	6	4	1	-7539.17	356594	315.2	-6872.51	287757	600.0
29	9	3	3	-7599.88	288322	265.3	-6876.98	278023	600.0
30	8	3	3	-7647.50	367318	429.2	-7080.80	278160	600.0
Average over all runs:				-7612.74	312879	313.7	-6941.14	255498	600.0

Table 7: 30 runs by MIDACO (max time = 7200) + SQP (max iter=1000)

	Booster-Config.			SQP			MIDACO		
Run	B_1	T_1	T_2	$f(x,y)$	Eval	Time	$f(x,y)$	Eval	Time
1	9	3	1	-7599.88	357908	317.7	-7419.65	3455790	7200.0
2	9	3	1	-7599.88	353114	315.1	-7449.22	3450447	7200.0
3	8	3	3	-7647.50	363366	321.1	-7502.77	3443609	7200.0
4	9	3	1	-7599.88	267022	236.8	-7419.91	3449060	7200.0
5	9	3	5	-7599.88	309848	274.6	-7418.63	3460976	7200.0
6	9	3	1	-7599.88	173384	153.8	-7436.00	3472466	7200.0
7	9	3	1	-7599.88	346444	307.3	-7555.53	3456612	7200.0
8	9	3	4	-7599.88	265638	234.8	-7369.10	3457577	7200.0
9	7	3	3	-7567.75	6713	6.4	-7565.33	3445493	7200.0
10	9	3	4	-7599.88	284148	254.1	-7524.85	3445318	7200.0
11	8	3	3	-7524.57	7379	7.1	-7519.89	3447985	7200.0
12	8	3	3	-7647.50	354988	313.6	-7481.90	3459946	7200.0
13	9	3	1	-7599.88	270324	240.1	-7444.49	3453002	7200.0
14	8	3	3	-7647.50	363938	322.9	-7479.16	3451839	7200.0
15	9	3	5	-7599.88	266138	235.9	-7500.50	3464034	7200.0
16	9	3	5	-7599.88	301198	266.8	-7519.93	3481049	7200.0
17	9	3	3	-7599.88	344342	307.8	-7456.35	3450507	7200.0
18	9	3	1	-7599.88	273766	242.3	-7528.82	3454741	7200.0
19	9	3	1	-7599.88	298972	267.0	-7527.04	3458943	7200.0
20	9	3	4	-7599.88	324916	290.1	-7431.08	3468826	7200.0
21	9	3	5	-7599.88	355510	317.4	-7498.23	3487475	7200.0
22	9	3	5	-7599.88	341446	322.4	-7430.29	3042720	7200.0
23	8	3	3	-7647.43	309588	370.1	-7536.62	2879186	7200.0
24	9	3	5	-7599.88	349166	425.9	-7460.48	2435917	7200.0
25	8	3	3	-7513.12	7360	9.2	-7505.67	2568537	7200.0
26	6	4	1	-7539.17	361342	332.9	-7434.57	2871096	7200.0
27	9	3	5	-7599.88	313390	313.4	-7348.84	3060132	7200.0
28	9	3	3	-7599.88	263188	347.4	-7475.90	3150988	7200.0
29	8	3	3	-7647.50	355292	321.1	-7470.97	2873982	7200.0
30	8	3	3	-7647.50	365252	327.5	-7491.21	3336049	7200.0
Average over all runs:				-7600.91	285169.3	266.8	-7473.43	3294476.7	7200.0

In case of Table 6 MIDACO reveals the best known integer combination of $\{8, 3, 3\}$ in 13 out of 30 cases ($\sim$ 43 %). In 12 cases the integer combination of $\{9, 3, -\}$ has been found. Only one time (Run 11) an integer combination is found by MIDACO, that corresponds to a solution that is less optimal than the original combination of $\{6, 3, 3\}$. The average objective function value of the MIDACO solutions is 6941.14 kg corresponding to an average of 255498 function evaluation. The SQP algorithm is able to successfully refine all MIDACO solutions to the best known solutions presented in Table 5, requiring 312879 function evaluation and about 5 Minutes on average.

In case of Table 7 MIDACO reveals the best known integer combination of $\{8, 3, 3\}$ in 8 out of 30 cases ($\sim$ 27 %). In 20 cases the integer combination of $\{9, 3, -\}$ has been found. In all cases MIDACO reveals integer combination, that corresponds to solutions that are more attractive than the original combination of $\{6, 3, 3\}$. The average objective function value of the MIDACO solutions is 7473.43 kg corresponding to an average of 3294476 function evaluation. The SQP algorithm is able to successfully refine the MIDACO solutions to the best known solutions presented in Table 5 in 27 out of 30 cases. In three cases (Run 9, Run 11 and Run 25) the SQP algorithm stops prematurely.

Figure 1 contains plots regarding the best known solution to the multiple-stage launch vehicle problem corresponding to the integer combination $\{8, 3, 3\}$. This is in particular the altitude, control, velocity, mass, energy transfer and dynamic pressure progression of the launch vehicle

during its total flight. It can be seen, that the behavior is very similar to the original solution reported in GPOPS [7].

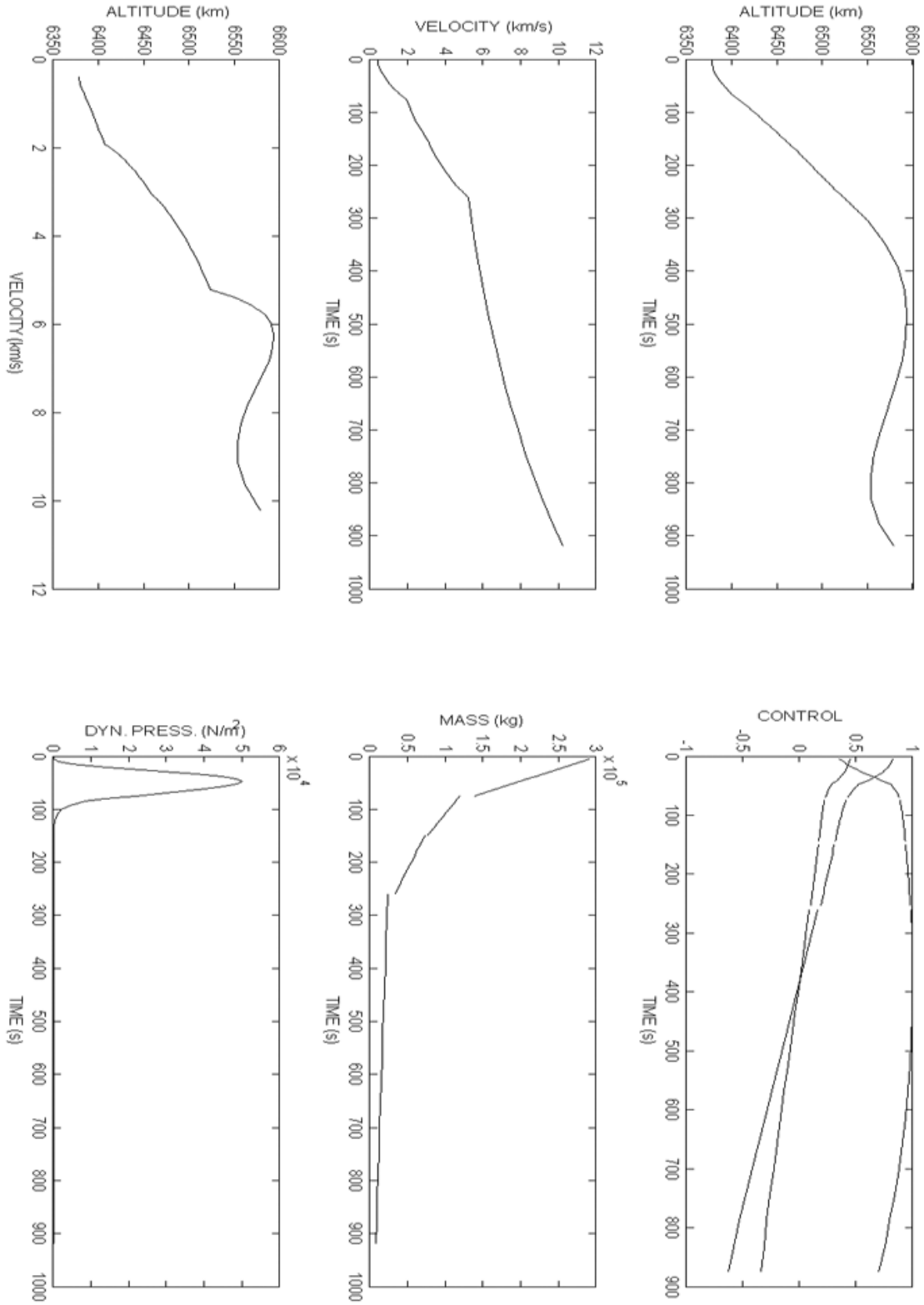

Figure 1: Illustration of the control and physical behavior of the launch vehicle

3.5 Launch Vehicle: Conclusions and Interpretation

The optimal control of a multiple-stage launch vehicle has been considered. In contrast to the original purely continuous approach in GPOPS [7], here some mixed integer extensions regarding the booster configuration along with further non-linear constraints have been added. Analyzing the impact of those discrete aspects (Table 5) revealed a non-intuitive and diverse integer complexity. The resulting MINLP problem can therefore be categorized as very challenging and consists of well over 100 variables and constraints, which is at the limit of current state of the art MINLP solvers, based on evolutionary algorithms.

Two numerical test series of 30 runs each applied a hybrid strategy of the stochastic MIDACO algorithm coupled with a deterministic SQP method to solve the MINLP problem. In both test series MIDACO was able to provide the best known integer combination of the MINLP with high probability, while the SQP method was in most cases able to successfully refine those MIDACO solutions in a reasonable time of about 5 Minutes. Interestingly it could be observed, that a relative small cpu-time budget of 600 (10 Minutes) seconds for MIDACO was sufficient, to obtain premature MINLP solutions of such quality, that the SQP algorithm could always successfully refine those. In contrast to this, the larger budget of 7200 seconds had two undesired effects, which is a less probability in the best known integer combination and some over-fitting that led to premature convergence of the SQP method. The lower probability in finding the best known integer seems to be correlated to the amount of $\{9, 3, -\}$ combinations revealed by MIDACO. Due to the insignificance of the third integer variable T_2 in the scenario of $B_1 = 0$, those solutions have a five times higher probability. As the refinement of the solution to the optimal control problem is highly depending on the precision of the continuous variables, a longer runtime for MIDACO implies therefore a higher probability to switch to the sub-optimal $\{9, 3, -\}$.

4 Interplanetary Space Mission Design

The design of an interplanetary space mission from Earth to Jupiter is considered here. This application is based on the real world mission *Galileo* launched by NASA in October 1989 (see http://solarsystem.nasa.gov/galileo/). The Galileo mission was the first one to implement *gravity assist* maneuvers, where the direction and velocity of the spacecraft is changed due to the gravitational force of a planet. The Galileo probe performed three flybys in total (one at Venus and two at Earth) and several minor flybys at asteroids on its way to Jupiter. Here a general interplanetary space mission model is assumed, considering the flyby planets as discrete optimization variables. The model setup is so general, that it will allow several possible feasible trajectories from Earth to Jupiter, including the Galileo type of mission. Together with the continuous optimization parameters (e.g. for thrusting and flyby altitudes), the mission design forms a challenging MINLP problem. To the best knowledge of the author, this is the first time, that a interplanetary space trajectory optimization problem is considered as MINLP.

4.1 Space Mission Layout

The space mission is implemented as an optimal control problem containing several stages, corresponding to different *arcs* of the mission. Here an *arc* describes the time between two major events of the mission, such as flyby or thrusting maneuvers. The mission is characterized by three flyby maneuvers, which is based on the real Galileo mission. For thrusting it assumes one *deep space maneuver* (DSM), which is a thrusting maneuver that happens in space where the influence of any planet can be neglected. Further an escape (from Earth) and a capture (for Jupiter) thrusting maneuver are supposed. Further the flyby altitudes are assumed as continuous optimization variables as well as the time duration of each arc. Figure 2 illustrates the mission layout regarding the five arcs and major events.

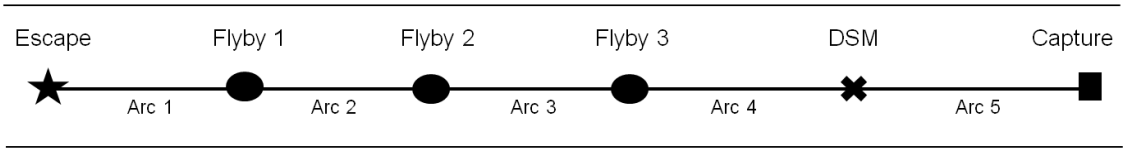

Figure 2: MGA-DSM space mission layout regarding arcs and major events

In total, the space mission model consists of 18 continuous optimization variables x and three integer variables y, representing the planet candidates for every flyby in the mission layout (see Fig. 2). All nine planets of the solar system are considered as possible flyby candidate. Thus the integer complexity of this MINLP formulation is $9^3 = 729$, whereas only the first four planets of the inner solar system seem to be reasonable flyby candidates. Table 8 lists the nine planets with their corresponding integer identification number.

Table 8: Planet numeration

Number	Planet
1	Mercury
2	Venus
3	Earth
4	Mars
5	Jupiter
6	Saturn
7	Uranus
8	Neptune
9	Pluto

Table 9 lists all optimization variables of the mission with a brief description and assumed lower and upper bounds. Note, that the three thrusting maneuvers are controlled in three cartesian dimensions (X, Y, Z). For gravity assists a minimal and maximal flyby altitude is assumed, which is based on the concrete flyby planet radius. The minimal flyby altitude Alt_{min} is defined as (101)% of the planet radius, where the 1% simulates the planet atmosphere. Only for the Earth, an atmosphere of 300km (instead of 63.78km, which would be 1% of the Earth radius) is assumed in particular, to take into account satellites orbiting the Earth. The maximal flyby altitude Alt_{max} is defined as $Alt_{max} = 100 \cdot Alt_{min}$, which also allows to model flybys of very little impact on the trajectory. The launch date of the space mission is considered a further continuous optimization variable, where a launch time window of two years (from 1 Jan. 1989 to 31 Dec. 1990) is assumed.

Table 9: Optimization variables x (continuous) and y (integer) with bounds

Variable	Description	Lower Bound	Upper Bound
continuous			
x_1	Launch Date	0 (01 Jan. 1989)	730 (31 Dec. 1990)
x_2	Duration of Arc 1	0 (days)	200 (days)
x_3	Duration of Arc 2	0 (days)	400 (days)
x_4	Duration of Arc 3	0 (days)	800 (days)
x_5	Duration of Arc 4	0 (days)	100 (days)
x_6	Duration of Arc 5	0 (days)	1200 (days)
x_7	Thrust Escape (X direction)	-6000.0 (m/sec)	6000.0 (m/sec)
x_8	Thrust Escape (Y direction)	-6000.0 (m/sec)	6000.0 (m/sec)
x_9	Thrust Escape (Z direction)	-3000.0 (m/sec)	3000.0 (m/sec)
x_{10}	Thrust Capture (X direction)	-6000.0 (m/sec)	6000.0 (m/sec)
x_{11}	Thrust Capture (Y direction)	-6000.0 (m/sec)	6000.0 (m/sec)
x_{12}	Thrust Capture (Z direction)	-3000.0 (m/sec)	3000.0 (m/sec)
x_{13}	Thrust DSM (X direction)	-1000.0 (m/sec)	1000.0 (m/sec)
x_{14}	Thrust DSM (Y direction)	-1000.0 (m/sec)	1000.0 (m/sec)
x_{15}	Thrust DSM (Z direction)	-500.0 (m/sec)	500.0 (m/sec)
x_{16}	Altitude Flyby 1	0.00 ($\sim Alt_{min}$)	1.00 ($\sim Alt_{max}$)
x_{17}	Altitude Flyby 2	0.00 ($\sim Alt_{min}$)	1.00 ($\sim Alt_{max}$)
x_{18}	Altitude Flyby 3	0.00 ($\sim Alt_{min}$)	1.00 ($\sim Alt_{max}$)
integer			
y_1	Planet Flyby 1	1 (Mercury)	9 (Pluto)
y_2	Planet Flyby 2	1 (Mercury)	9 (Pluto)
y_3	Planet Flyby 3	1 (Mercury)	9 (Pluto)

The objective function $f(x, y)$ to be minimized is defined as the total ΔV (*change in velocity*) of the mission, which is produced by all the individual thrusting maneuvers. This is in particular the ΔV_{escape}, $\Delta V_{capture}$ and ΔV_{DSM}. While the ΔV_{DSM} directly corresponds to the thrusting of the DSM, the ΔV_{escape} and $\Delta V_{capture}$ take into account velocity of the escape and capture planet and thus express the velocity change in reference to the planet and not the Sun. The mathematical formulation of the objective function is given in Equation 17:

$$
\begin{aligned}
f(x,y) &= \Delta V = \Delta V_{escape} + \Delta V_{capture} + \Delta V_{DSM}, \\
&\text{with} \\
\Delta V_{escape} &= \sqrt{\frac{2\mu^{Earth}}{x_7^2 + x_8^2 + x_9^2}} - \sqrt{2\mu^{Earth}\left(\frac{1}{R_{peregee}^{Earth}} - \frac{1}{R_{apogee}^{Earth} - R_{peregee}^{Earth}}\right)}, \\
\Delta V_{capture} &= \sqrt{\frac{2\mu^{Jupiter}}{x_{10}^2 + x_{11}^2 + x_{12}^2}} - \sqrt{2\mu^{Jupiter}\left(\frac{1}{R_{peregee}^{Jupiter}} - \frac{1}{R_{apogee}^{Jupiter} - R_{peregee}^{Jupiter}}\right)}, \\
\Delta V_{DSM} &= \sqrt{x_{13}^2 + x_{14}^2 + x_{15}^2}.
\end{aligned}
\tag{17}
$$

The astrophysical constants used in Equation 17 are listed in Table 10.

Table 10: Gravitation parameter and apsis for Earth and Jupiter

$\mu^{Earth} =$	$3.986 \cdot 10^{14}$	$R_{peregee}^{Earth} =$	6778000	$R_{apogee}^{Earth} =$	42165000
$\mu^{Jupiter} =$	$1.267 \cdot 10^{17}$	$R_{peregee}^{Jupiter} =$	10^9	$R_{apogee}^{Jupiter} =$	$2 \cdot 10^{10}$

The mission is restricted to twelve nonlinear constraints $g_1(x, y), ..., g_{12}(x, y)$ which take into account on the trajectory of the spacecraft. The calculation of the trajectory (nominated as $R^{Spacecraft}(t)$) is not given in detail here, as its implementation takes into account several complex subroutines, which were friendly provided by EADS Astrium (http://www.astrium.eads.net/).

Those subroutines model for example the orbit propagation of the spacecraft based on Lagrange coefficients. Here only the mathematical structure of the constraints $g(x, y)$ should be illustrated, which form the MINLP problem. The full implementation of the Mission (including the subroutines provided by Astrium) can be downloaded at http://www.midaco-solver.com/applications.html for studying purposes. Table 11 describes the notation used in the constraints for those functions, which are not given in detail here.

Table 11: Notation for constraints

Notation	Description
$R^{Spacecraft}(t)$	Position of the spacecraft at time t
$R^{Planet}(t, P)$	Position of planet P at time t
$V^{Planet}(t, P)$	Velocity of planet P at time t
$V_X^{Spacecraft}(t)$	Velocity of the spacecraft (X direction)
$V_Y^{Spacecraft}(t)$	Velocity of the spacecraft (Y direction)
$V_Z^{Spacecraft}(t)$	Velocity of the spacecraft (Z direction)
$rota^{Planet}(P)$	Orbit rotation time (in days) of planet P
$S^{\%}$	Sphere of action radius of a planet (given in percentage $\in [0, 1]$)

The first three constraints model the necessary condition for the flyby maneuvers, that the spacecraft is within the *sphere of action* (roughly speaking the position) of the concrete flyby planet (here y_1, y_2 and y_3) at the time, when the actual flyby maneuver is supposed to happen. This time point depends on the launch date x_1 and the time durations of the arcs $(x_2, ..., x_6)$. The radius of the sphere of action is based on the distance of the concrete flyby planet to the sun. Here the constraints assume, that the spacecraft is within at least $S^{\%}$ percent (with $S^{\%} \in [0, 1]$, where $0 \sim 0\%$ and $1 \sim 100\%$) of this distance, which is considered the *sphere of action*. Equation 18 states the mathematical structure of the first three constraints:

$$\begin{aligned} g_1(x, y) =& \|R^{Spacecraft}(x_1 + x_2) - R^{Planet}(x_1 + x_2,\ y_1)\| \\ & \leq S^{\%}\ \|R^{Planet}(x_1 + x_2,\ y_1)\|, \\ \\ g_2(x, y) =& \|R^{Spacecraft}(x_1 + x_2 + x_3) - R^{Planet}(x_1 + x_2 + x_3,\ y_2)\| \\ & \leq S^{\%}\ \|R^{Planet}(x_1 + x_2 + x_3,\ y_2)\|, \\ \\ g_3(x, y) =& \|R^{Spacecraft}(x_1 + x_2 + x_3 + x_4) - R^{Planet}(x_1 + x_2 + x_3 + x_4,\ y_3)\| \\ & \leq S^{\%}\ \|R^{Planet}(x_1 + x_2 + x_3 + x_4,\ y_3)\|. \end{aligned} \tag{18}$$

The fourth constraint models the necessary condition, that the spacecraft is within the sphere of action of Jupiter (Planet number 5) at the very end of the mission. This is done analog to the first three constraints. Equation 19 states the mathematical structure of the fourth constraint:

$$\begin{aligned} g_4(x, y) =& \|R^{Spacecraft}(x_1 + x_2 + x_3 + x_4 + x_5 + x_6) \\ & - R^{Planet}(x_1 + x_2 + x_3 + x_4 + x_5 + x_6,\ 5)\| \\ & \leq S^{\%}\ \|R^{Planet}(x_1 + x_2 + x_3 + x_4 + x_5 + x_6,\ 5)\|. \end{aligned} \tag{19}$$

The fifth to seventh constraint model the necessary condition, that the velocity of the spacecraft at the very end of the mission is identical to the velocity (and direction) of the target planet, Jupiter. This is to later allow the spacecraft to orbit Jupiter. As the total velocity of Jupiter itself ($\|V^{Planet}(t, 5)\|$) is undirected, three constraints are necessary to take into account the specific directions (in X,Y, and Z). The same tolerance $S^{\%}$ for the *sphere of action* (described above) is applied here to the total velocity of the target planet, to allow some tolerance in fulfilling these

constraints. Equation 20 states the mathematical structure of the fifth to seventh constraint:

$$\begin{aligned} g_5(x,y) =& |V_X^{Spacecraft}(x_1+x_2+x_3+x_4+x_5+x_6) - \\ & V_X^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)| \\ & \leq S^{\%}\ \|V^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)\|, \\ \\ g_6(x,y) =& |V_Y^{Spacecraft}(x_1+x_2+x_3+x_4+x_5+x_6) - \\ & V_Y^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)| \\ & \leq S^{\%}\ \|V^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)\|, \\ \\ g_7(x,y) =& |V_Z^{Spacecraft}(x_1+x_2+x_3+x_4+x_5+x_6) - \\ & V_Z^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)| \\ & \leq S^{\%}\ \|V^{Planet}(x_1+x_2+x_3+x_4+x_5+x_6,\ 5)\|. \end{aligned} \tag{20}$$

Some additional constraints are imposed, to avoid feasible, but undesired solutions. Those undesired solutions might attract the optimization algorithm, but are not of any relevance and should therefore be avoided. The eighth constraint impose, that not all flyby maneuvers happen at the planet Earth. Equation 21 states the mathematical structure of the eighth constraint:

$$g_8(x,y) = \begin{cases} \infty & \text{, if } y_1 = y_2 = y_3, \\ 0 & \text{, else.} \end{cases} \tag{21}$$

The ninth to twelfth constraint impose, that if two successive flybys happen at the same planet, the time duration between those flybys must be at least greater or equal than half of the total rotation time of the flyby planet (in respect to the sun). Equation 22 states the mathematical structure of the ninth to twelfth constraint constraint:

$$\begin{aligned} g_9(x,y) &= \begin{cases} \frac{rota^{Planet}(y_1)}{2} - x_2 \leq 0 & \text{, if } y_1 = 3, \\ 0 & \text{, else,} \end{cases} \\ \\ g_{10}(x,y) &= \begin{cases} \frac{rota^{Planet}(y_2)}{2} - x_3 \leq 0 & \text{, if } y_2 = y_1, \\ 0 & \text{, else,} \end{cases} \\ \\ g_{11}(x,y) &= \begin{cases} \frac{rota^{Planet}(y_3)}{2} - x_4 \leq 0 & \text{, if } y_3 = y_2, \\ 0 & \text{, else,} \end{cases} \\ \\ g_{12}(x,y) &= \begin{cases} \frac{rota^{Planet}(5)}{2} - (x_5+x_6) \leq 0 & \text{, if } 5 = y_3, \\ 0 & \text{, else.} \end{cases} \end{aligned} \tag{22}$$

4.2 Numerical Results

MIDACO has been used to solve this application in a two step approach. In a first step, a number of several test runs using different random seeds are performed on the full mission model using a moderate parameter of $S^{\%} = 0.03$ (which is 3%) for the accuracy of the sphere of action around the planets. As the accuracy of the sphere of action is a crucial parameter in the model, this moderate accuracy will allow MIDACO to identify possible feasible mission trajectories more easily, while concentrating more on the integer complexity of the problem. In a second step, the most promising solutions found within the first step should then be further refined by MIDACO, assuming a higher accuracy of 0.5% for the sphere of action.

Table 13 lists the results of the first optimization step, where MIDACO is applied 10 times on the space mission model using different random seeds and a moderate precision of 3% for the sphere of

action in the model. Every test run was performed for a duration of one hour on a PC with an Intel Xeon CPU E5640 (2.67GHz clock rate, 4GB RAM) which corresponds to some hundred million function evaluation. Note, that such a high amount of function evaluation is nothing unusual for a stochastic algorithm applied on a difficult problem. As the MIDACO software is capable of processing millions of iterates in seconds, the total cpu runtime of one hour remains here still reasonable.

Table 12: Abbreviations for Table 13

Abbreviation	Explanation
Run	Number of test run
Launch	Date for mission departure from Earth
ΔV	Objective function value (m/sec)
Duration	Duration of total mission (Years)
FlyBy 1	Planet selected by MIDACO for 1st gravity assist maneuver
FlyBy 2	Planet selected by MIDACO for 2nd gravity assist maneuver
FlyBy 3	Planet selected by MIDACO for 3rd gravity assist maneuver

Table 12 explains the abbreviations used in Table 13.

Table 13: 10 test runs by MIDACO on mission model with 3% sphere of action

Run	Launch	ΔV	Duration	FlyBy 1	FlyBy 2	FlyBy 3
1	6 Nov. 1989	2553	5.88	Venus	Earth	Earth
2	30 Nov. 1989	3310	4.83	Venus	Earth	Earth
3	30 Nov. 1989	3218	4.88	Venus	Earth	Earth
4	10 Jul. 1989	3390	3.97	Venus	Earth	Mars
5	20 Nov. 1989	2890	4.77	Venus	Earth	Earth
6	23 May 1989	2759	5.35	Earth	Venus	Earth
7	21 Mar 1989	*infeasible*	4.85	Earth	Earth	Mars
8	13 Apr. 1989	3290	4.54	Earth	Venus	Earth
9	30 Nov. 1989	3289	4.79	Venus	Earth	Earth
10	16 Sep. 1989	2684	6.10	Venus	Earth	Earth

As it can be seen from Table 13, MIDACO did reveal a number of possible mission trajectory, based on different flyby planet candidates. Those missions vary strongly in their characteristics, as it can be seen from the differences in the launch date, objective function values ΔV and total flight durations. Only in one case (test run number 7), MIDACO did not succeed to find a feasible mission trajectory. The integer combination mostly attracted by MIDACO is (Venus, Earth, Earth), which is indeed the same combination as used in the original Galileo mission. Besides this combination, the combination (Venus, Earth, Mars) found in test run number 4 seems somewhat interesting. This mission has the worst (in esp. highest) objective function value, but also the shortest flight duration.

In a second optimization step, the solutions corresponding to test run number 1 (named Mission1) and number 4 (named Mission4)from Table 13 should now be refined by MIDACO, assuming a more precise accuracy of 0.5% for the sphere of action around the planets. For this purpose, those solutions are given to MIDACO as starting point and the *QSTART* (using a value of 10000, see Section 2.1 or [13]) is activated for a more efficient search in the vicinity of the submitted initial solution. MIDACO is applied to the refined model for one hour again for both initial solutions (in esp. Mission1 and Mission4 from Table 13). Table 14 shows the solutions of the refinement of Mission1 and Mission4 with details on the individual flyby maneuvers. Table 14 also compares these two missions generated by MIDACO with the original Galileo mission regarding their main characteristics.

Table 14: Comparison between original Galileo and MIDACO Missions

	Galileo Mission	Mission1 refine 0.5 %	Mission4 refine 0.5 %
Launch	18 Oct. 1989	8 Nov. 1989	6 Jul. 1989
Duration	6.14 Years	6.14 Years	4.15 Years
ΔV	*unknown*	3,350 m/sec	5,177 m/sec
1st Flyby			
Planet	Venus	Venus	Venus
Date	10 Feb. 1990	23 Feb. 1990	21 Jan. 1990
Altitude	16,000km	28,901km	3,013km
2nd Flyby			
Planet	Earth	Earth	Earth
Date	8 Dec. 1990	5 Dec. 1990	4 Sep. 1990
Altitude	960km	473,191km	1,754km
3rd Flyby			
Planet	Earth	Earth	Mars
Date	8 Dec. 1992	4 Dec. 1992	31 Dec. 1990
Altitude	303km	300km	39km

Analyzing the Missions from Table 14, the structural difference between Mission1 and Mission4 is evident. Mission4 is however in so far interesting in respect to Mission1, as it provides a much shorter ($\sim$ 32.4%) flight duration to the price of an equivalent increased ($\sim$ 35.3%) objective function value. More interestingly is the striking similarity between Mission1 and the original Galileo mission. With a shift of about 22 days regarding the launch date, these two mission share exactly the same flight duration and are closely related regarding all the characteristics of their gravity assist maneuvers, except the flyby altitude at the 2nd flyby. Here a significant difference between 960km (Galileo) and 473,191 km (Mission1) occurs. However, this difference can be well explained by taking into account, that for the original Galileo mission the observation of asteroids were an objective, while asteroids were not considered in the model formulation here. Both missions perform their first Earth flyby in December 1990. The Galileo mission performs its first flyby at Earth at a moderate altitude of 960km in order to significantly increase the semi major axis of its orbit to visit the Gaspra asteroid in October 1991. This new orbit is then rotated one time by the Galileo probe, before it performs its second flyby at Earth two years later in 1992. As in this model here asteroids were not considered, this maneuver is not a target in Mission1. Therefore Mission1 performs its first flyby at Earth at a very large altitude of 473,191 km in order to have only a very small gravitational impact on its trajectory. This way Mission1 can perform two rotations of its near Earth orbit in a row, before it performs its second Earth flyby two years later in December 1992 like the original Galileo mission.

Figure 3 shows the space mission trajectory of the original Galileo mission (image taken from *Wikimedia* (http://commons.wikimedia.org)) and the Mission1 generated by MIDACO. The coincidence of the major events of both missions can be well observed. Also the difference in the trajectories between the 2nd and 3rd flyby can be seen: While Galileo performs one orbit rotation visiting Gaspra, Mission1 remains in a close Earth orbit rotating it two times.

Please note, that **Video Animations** of the MIDACO Mission1 and Mission4 are available at http://www.midaco-solver.com/space.html for downloading.

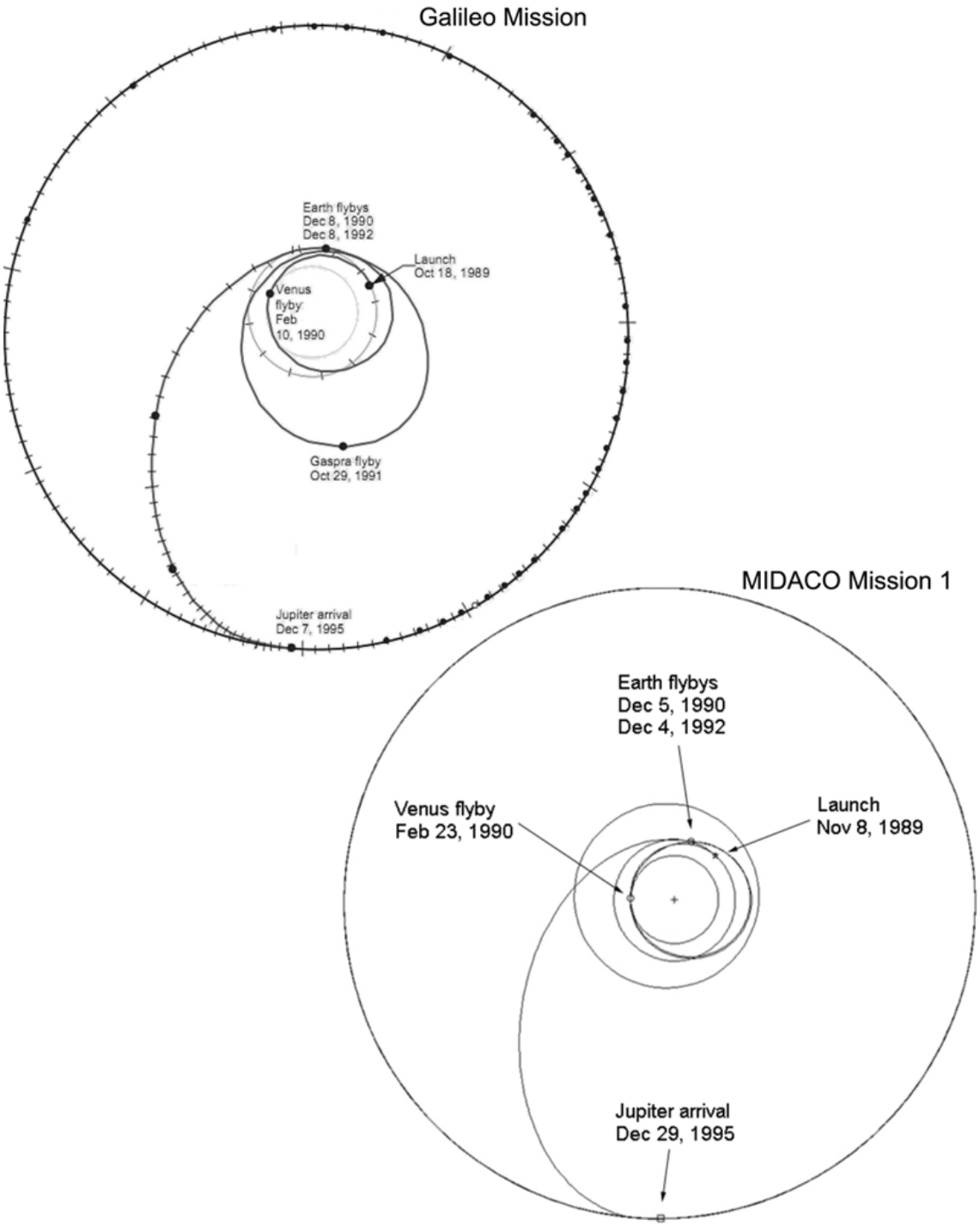

Figure 3: Space trajectories of the NASA Galileo mission and MIDACO Mission1

4.3 Space Mission Design: Conclusions

For the first time a multi gravity assist interplanetary space mission was considered in an MINLP formulation, considering flyby planet candidates as discrete decision variables. As space trajectory optimization problems are known to be very difficult (see Section [2]), this MINLP approach can be considered as exceptionally challenging. In accordance to the Galileo mission by NASA in 1989, a general space mission model for transfers from Earth to Jupiter was formulated, based on three

flyby maneuvers. In a two step optimization process, it could be shown, that MIDACO is able to generate feasible space trajectories in a reasonable time and accuracy fully automatically. The results from Table 13 indicate, that the space mission model setting was general enough, to allow several feasible space trajectories (which implies a sufficient large search space). Among some different space mission trajectory candidates, MIDACO did reveal the Galileo type of mission with high probability (7 out of 10 cases). The accuracy of the sphere of action around the planets was identified as crucial model parameter. Within a second optimization step, MIDACO was able to refine its generated missions to a sufficient accuracy of 0.5% and the fully automatically generated mission by MIDACO showed an intriguing coincidence with the characteristics of the original Galileo mission from 1989.

5 Optimal Control of an F8-Aircraft Manoeuvre

Here the optimal control of a simplified aircraft manoeuvre is discussed. This application is known as the F-8 aircraft control problem introduced by Kaya and Noakes [4]. Here we refer to a formulation of this application that is available from *mintOC* [8] at `http://mintoc.de/index.php/F-8_aircraft`. Several reference solutions to this application can be found at the *mintOC* [8] webpage. Among those are solutions obtained by well known solvers such as *BONMIN*, *KNITRO* and *IPOPT*. Please note, that none of the above mentioned solvers is capable to solve this application to its current best known solution, due to the information given on *mintOC* [8]. Hence we consider this as a challenging application with good possibilities to compare the solution quality obtained by the here considered approach with concurrent ones.

The objective of this application is the minimization of the final time of a simplified aircraft manoeuvre. The manoeuvre is performed using a bang-bang control approach, thus the control can only switch between two values. Here this application is formulated as a NLP optimal control problem, applying 6 different stages, whereas every stage represents a switch in the bang-bang control. This implies 6 optimization variables, which correspond to the starting time points of the stages, defining the switching time point of the bang bang control. This approach coincides with the representation of the reference solutions given at *mintOC* [8], where the optimal control stages are referred to as arcs. Besides the 6 optimization variables, 3 equality constraints must be fulfilled at the end of the manoeuvre, representing the correct final state of the aircraft. This is in especially that all three differential states must be zero at the final time of the manoeuvre.

Equation 23 expresses the F8 aircraft manoeuvre regarding the three time depended considered differential states $\dot{x}_0$, $\dot{x}_1$ and $\dot{x}_2$. The initial state $x(0)$ and the final state $x(t_{final})$ are also stated, whereas the final state implies the above mentioned three equality constraints.

$$
\begin{aligned}
\text{Minimize} \quad & t_{final}, \\
\text{s.t.:} \quad \dot{x}_0 = & -0.877x_0 + x_2 - 0.088x_0x_2 + 0.47x_0^2 - 0.019x_1^2 - x_0^2x_2 + 3.846x_0^3, \\
& -(0.215\xi - 0.28x_0^2 - 0.47x_0\xi^2 - 0.63\xi^3)w, \\
& -(-0.215\xi + 0.28x_0^2\xi - 0.47x_0\xi^2 + 0.063\xi^3)(1-w), \\
\dot{x}_1 = & \; x_2, \\
\dot{x}_2 = & -4.208x_0 - 0.396x_2 - 0.47x_0^2 - 3.564x_0^3, \\
& -(20.967\xi - 6.265x_0^2\xi - 46x_0\xi^2 - 61.4\xi^3)w, \\
& -(-20.967\xi + 6.265x_0^2\xi - 46x_0\xi^2 + 61.4\xi^3)(1-w), \\
x(0) = & \; (0.4655, 0, 0)^T, \\
x(t_{final}) = & \; (0,0,0)^T.
\end{aligned}
\tag{23}
$$

The bang-bang control $w \in \{0,1\}$ is defined in Equation 24. The bang-bang structure is here defined by the time points $t_1,\ t_2,\ t_3,\ t_4,\ t_5$ and t_{final}, marking the beginning or end of a different

stage (also called *arc*). These six time points are the decision variables, whereas the last one (t_{final}) also expresses the objective function.

$$w(t) = \begin{cases} 1 & ,\text{if } 0 \leq t < t_1, \\ 0 & ,\text{if } t_1 \leq t < t_2, \\ 1 & ,\text{if } t_2 \leq t < t_3, \\ 0 & ,\text{if } t_3 \leq t < t_4, \\ 1 & ,\text{if } t_4 \leq t < t_5, \\ 0 & ,\text{if } t_5 \leq t \leq t_{final}. \end{cases} \tag{24}$$

The F8-Aircraft application is solved by a combination of MIDACO and a SQP algorithm (see Section 2.1). MIDACO is first applied on the optimal control problem using the lower bounds (zero) as starting point, whereas the SQP algorithm is then afterwards called, applying the MIDACO solution as starting point. Two different setups for MIDACO are assumed: i) using default parameters and ii) using tuned parameters regarding the algorithm (in esp. *Qstart* and *Oracle*, see Section 2.1 or [13]). For the **default setup**, MIDACO is given a maximal time budget of 600 seconds, or stops before this limit by its own automatic stopping criteria (in esp. $Autostop = 5$, see Section 2.1 or [13]). For the **tuned setup**, MIDACO is given a maximal time budget of 60 seconds, a *Qstart* parameter of 100 (in order to focus the search process)and an *Oracle* parameter of 4.0 (which is based in the best known solution corresponding to an objective function value of 3.78, see Table 17). MIDACO assumes a moderate accuracy of 10^{-2} for the constraint violation, whereas the SQP algorithm assumes a higher accuracy of 10^{-6}. The idea behind this approach is that MIDACO delivers a reasonable good starting point for the SQP algorithm, which then returns a highly accurate solution. Table 15 and Table 16 shows the numerical results by this approach for 10 test runs (using a different random seed each time) by MIDACO using default parameters and tuned ones respectively. All runs were performed on a computer with an Intel(R) Core(TM) i7 Q820 CPU with 1.73GHz clock rate and 4GB RAM.

Table 15: Results of 10 test runs on F8-Aircraft using MIDACO (**default**) and SQP

	MIDACO			SQP		
Test Run	Objective	Evaluation	Time	Objective	Evaluation	Time
1	4.017191	575038	127.34	3.780211	106	0.04
2	3.753558	467211	126.49	3.780212	184	0.09
3	6.830677	313259	82.77	6.827373	221	0.13
4	3.740785	904630	262.50	3.780211	210	0.09
5	6.873262	1145671	304.39	6.827373	299	0.18
6	3.735261	422985	101.00	3.780211	73	0.03
7	6.860254	2097135	568.07	3.780211	711	0.44
8	3.766495	329588	89.35	3.780211	313	0.14
9	6.288805	1239587	600.00	6.322984	104	0.06
10	4.573276	667050	227.74	3.780211	3274	1.95
Average:	5.043956	820663	248.96	4.643921	549	0.32

Table 16: Results of 10 test runs on F8-Aircraft using MIDACO (**tuned**) and SQP

	MIDACO			SQP		
Test Run	Objective	Evaluation	Time	Objective	Evaluation	Time
1	3.775308	107930	60.00	3.780211	203	0.13
2	3.781723	133881	60.00	3.780211	170	0.07
3	3.735079	109770	60.00	3.780211	63	0.04
4	3.918386	104279	60.00	3.780211	206	0.14
5	4.092283	102022	60.00	3.780211	207	0.14
6	4.020482	107319	60.00	3.780211	92	0.06
7	3.733805	114478	60.00	3.780211	63	0.04
8	3.797692	100179	60.00	3.780211	157	0.11
9	4.048429	101560	60.00	3.780212	316	0.20
10	3.786220	121182	60.00	3.780211	108	0.08
Average:	3.868941	110260	60.00	3.780211	158	0.10

Table 15 displays the best solution found by the combination of MIDACO and SQP from Table 17 and compares it with the best known solution (found by Sager) from *mintOC* [8].

Table 17: F-8 Aircaft control problem solutions

Arc	w(t)	Sager	MIDACO + SQP
1	1	1.13492	1.1368996475
2	0	0.34703	0.3457308852
3	1	1.60721	1.6071985027
4	0	0.69169	0.6048388008
5	1	0	0.0000000000
6	0	0	0.0855435144
Infeasibility	-	2.21723e-07	0.4896e-13
Objective	-	3.78086	3.780211

Figure 4 displays the three differential states of the aircraft model over time, corresponding to the best solution presented in Table 17.

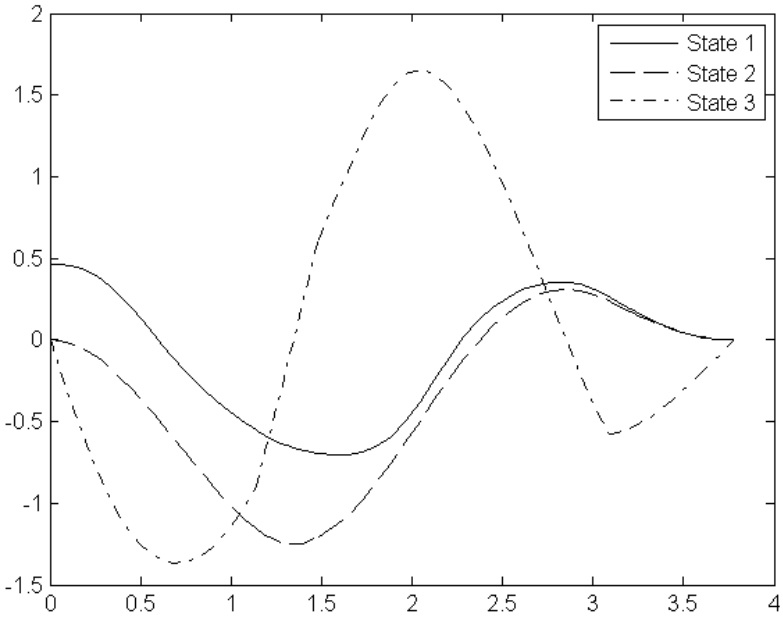

Figure 4: Differential states corresponding to best known solution

From the results from Table 15 it can be seen, that MIDACO (using **default** parameters) delivers four out of ten times a solution close to the best known one. In two cases MIDACO delivers a moderate solution quality of an objective function value between 4.01 and 4.57, while in the remaining four cases a worse objective function value between 6.28 and 6.87 is revealed. The SQP algorithm is able to refine the MIDACO solutions to the best known one (corresponding to an objective function value of 3.78) in seven out of ten cases to a high accuracy in less than a second.

From the results from Table 16 it can be seen, that MIDACO (using **tuned** parameters) delivers in all cases good solutions (in especially those with an objective function value smaller or equal to the predefined $Oracle = 4.0$). Note that the cpu time budget for MIDACO in Table 16 is only 60 instead of 600 seconds like in Table 15. The SQP algorithm is able to refine the MIDACO solutions to the best known one in all cases. Comparing the MIDACO results from Table 15 and Table 16 demonstrate the possible performance gain in tuning the algorithmic parameters and the effectiveness of the oracle penalty method (see [12]). In total it can be concluded, that the proposed hybrid approach here is well capable to robustly solve this application in a reasonable time with high accuracy.

6 Conclusions

Two space and one aerospace application have been presented and solved here by MIDACO. Taking advantage of the MINLP capabilities of the global optimization software MIDACO, the two space applications have been formulated as mixed integer problems, which is still a challenging novelty in space engineering. It could be shown, that MIDACO does robustly solve the presented applications in a reasonable cpu time. Furthermore it could be demonstrated, that modelling the applications as mixed integer problems offers an intriguing surplus in either an improved objective function (see Section 3) or general model capabilities (see Section 4). Additionally to the novel MINLP formulations, the possible performance gain by a hybridization of the stochastic MIDACO algorithm with a deterministic SQP algorithm has been investigated in two out of the three applications. It could be shown, that this approach is able to deliver highly accurate solutions and significantly shortens the required cpu time (see Table 6 and Table 15). It total it can be concluded, that MINLP is an ambitious and promising approach in (aero)space engineering and the recently developed MIDACO software (and its hybridization with local optimization algorithms) is well capable in solving such applications.

Acknowledgments

The authors would like to acknowledge the supported by the project *"Non-linear mixed-integer-based Optimisation Technique for Space Applications" (ESTEC/Contract No. 21943/08/NL/ST)* co-funded by ESA Networking Partnership Initiative, Astrium Limited (Stevenage, UK) and the School of Mathematics, University of Birmingham, UK.

References

[1] D. A. Benson. *A Gauss Pseudospectral Transcription for Optimal Control*. PhD thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 2004.

[2] European Space Agency (ESA) and Advanced Concepts Team (ACT). Gtop database - global optimisation trajectory problems and solutions. Software available at http://www.esa.int/gsp/ACT/inf/op/globopt.htm, 2011.

[3] G. T. Huntington. *Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control Problems*. PhD thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 2007.

[4] Y. Kaya, C. and L. Noakes, J. Computational algorithm for time-optimal switching control. *J. Optimiz. Theory App.*, 117(1):69–92, 2003.

[5] G.R. Kocis and I.E. Grossmann. Global optimization of nonconvex minlp problems in process synthesis. *Ind. Eng. Chem.*, 27:1407–1421, 1988.

[6] G. Maria, X. Zu, and J. Sun. Multi-objective minlp optimization used to identify theoretical gene knockout strategies for e. coli cell. *Chem. Biochem. Eng. Q.*, 25(4):403–424, 2012.

[7] A. V. Rao, D. A. Benson, C. L. Darby, M. A. Patterson, C. Francolin, and G. T. Huntington. Algorithm 902: Gpops, a matlab software for solving multiple-phase optimal control problems using the gauss pseudospectral method. *ACM T. Math. Software*, 37(2):1–39, 2010.

[8] S. Sager. mintOC: Benchmark library of mixed-integer optimal control problems. Software available at http://mintoc.de, 2011.

[9] M. Schlueter. MIDACO - Global Optimization Software for Mixed Integer Nonlinear Programming. Software available at http://www.midaco-solver.com, 2011.

[10] M. Schlueter, J. A. Egea, and J. R. Banga. Extended ant colony optimization for non-convex mixed integer nonlinear programming. *Comput. Oper. Res.*, 36(7):2217–2229, 2009.

[11] M. Schlueter, J. A. Egea, Antelo L.T., Alonso A.A., and J. R. Banga. An extended ant colony optimization algorithm for integrated process and control system design. *Ind. Eng. Chem.*, 48(14):6723–6738, 2009.

[12] M. Schlueter and M Gerdts. The oracle penalty method. *J. Global Optim.*, 47(2):293–325, 2010.

[13] M. Schlueter, M. Gerdts, and Rueckmann J.J. A numerical study of MIDACO on 100 MINLP benchmarks. *Optimization, Taylor and Francis (accepted)*, 2012.