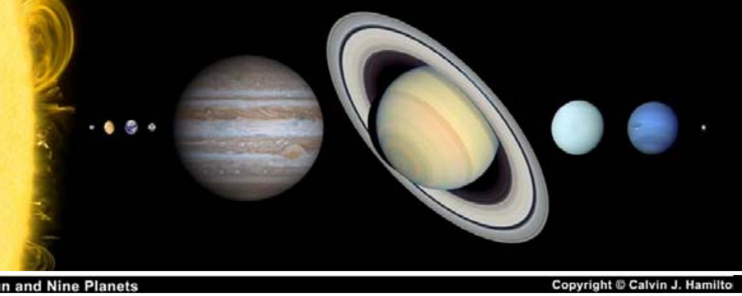


University of Reading

UNIVERSITY of GLASGOW

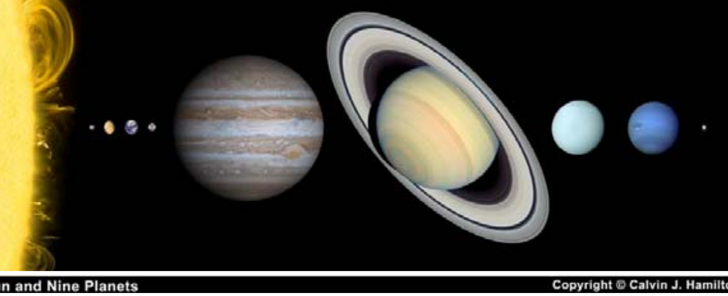
Global Trajectory Optimisation: Can we Prune the Solution Space when Considering Deep Space Manoeuvres?

Ariadna 06/4101

Final Meeting 10th September, 2007, ESTEC, Noordwjik, The Netherlands

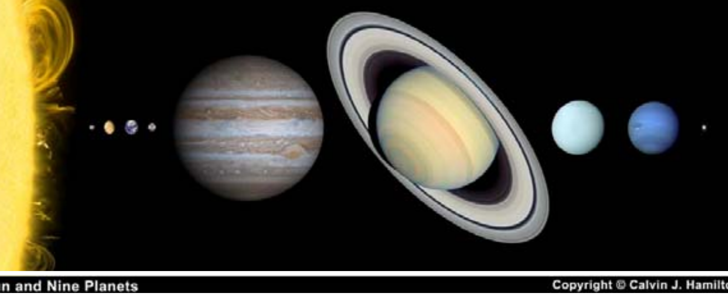
Massimiliano Vasile, Matteo Ceriotti
Department of Aerospace Engineering, University of Glasgow, Glasgow

Victor Becerra, Slawomir Nasuto
Department of Cybernetic, University of Reading



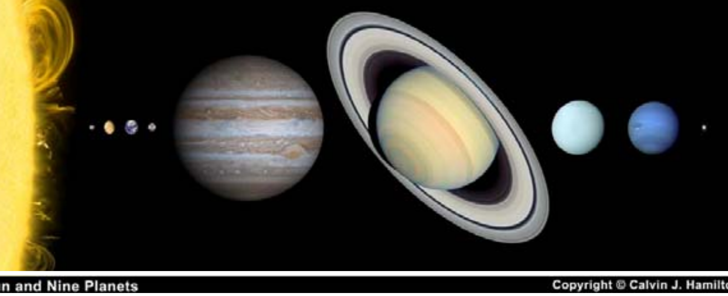
Outline

- Introduction to the Problem
- Problem Modeling
- Reading’s Solution Approach
- Glasgow’s Solution Approach
- Final Remarks



The Problem

- Two different problems of different complexity
  - Pruning the search space in the case of multiple deep space manoeuvres
  - Automated trajectory planning integrating transfer arcs of different nature



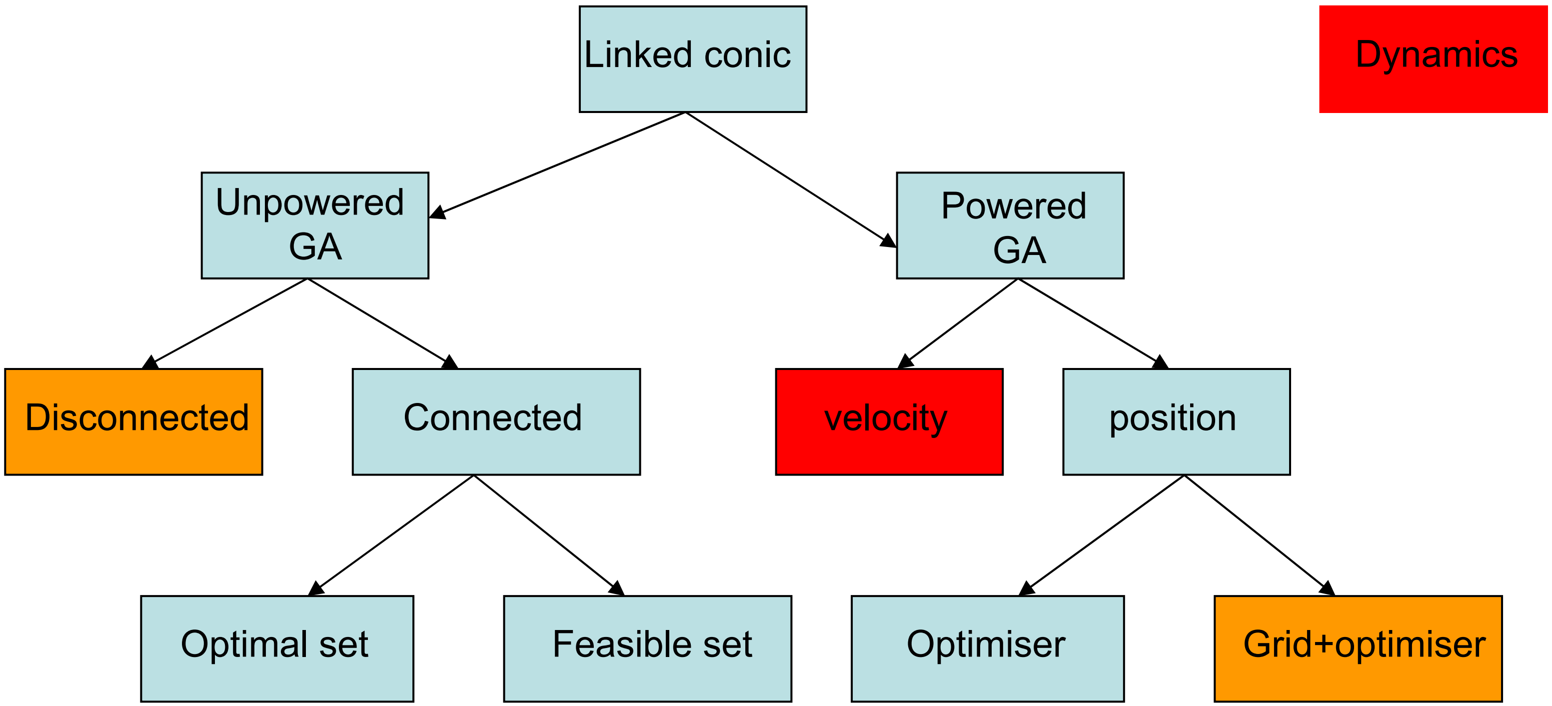
University of Reading

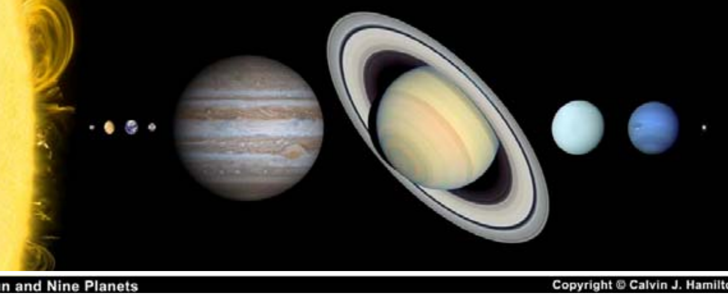
UNIVERSITY of GLASGOW

Available Solutions to the MGA problem

- S-TOUR and LTGAS-TOUR
  - 3D,MGA, DSMs, free sequence, systematic scan
- Swingby Calculator
  - 3D, MGA, DSMs, enumerative
- PAMSIT
  - 2D, MGA, no DSMs, free sequence, systematic search
- IMAGO
  - 3D, MGA, DSMs, free sequence, stochastic search
- DEIMOS
  - 3D, MGA, no DSMs, free sequence, systematic search
- GASP
  - 3D,MGA, no DSMs, systematic pruning plus GO
- MITRADES
  - 3D, MGA, DSMs

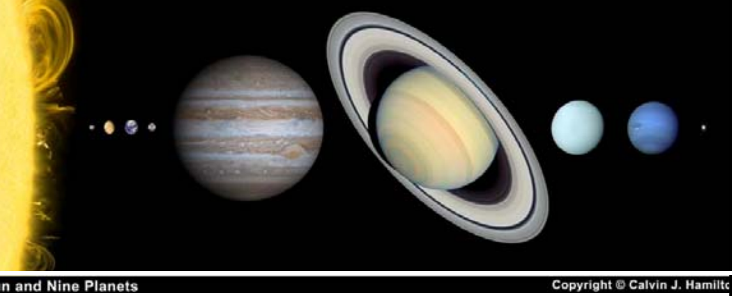
Possible Approaches to the MGA problem with DSMs



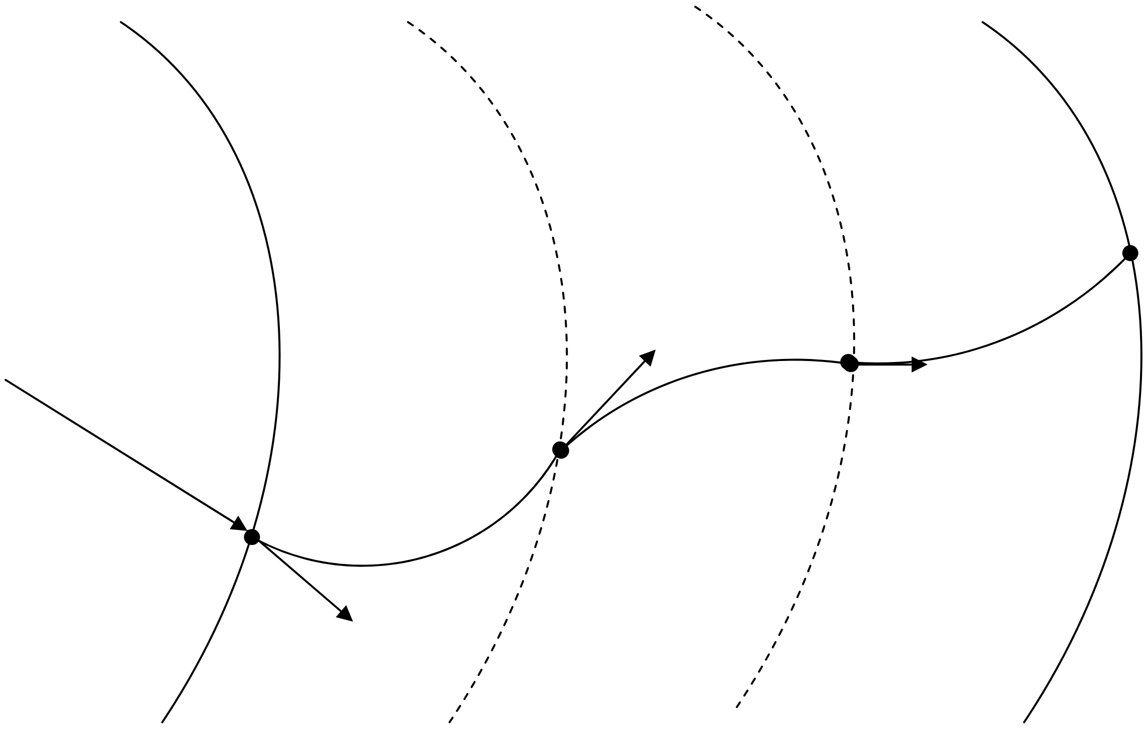


Trajectory Models

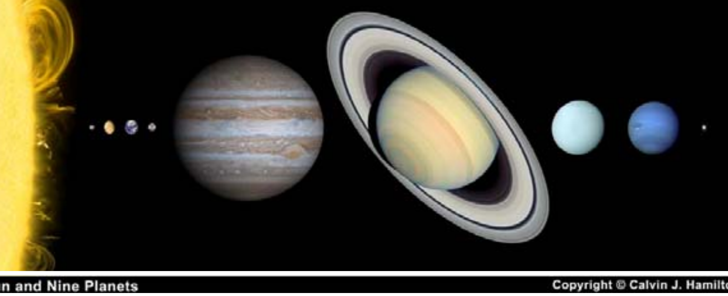
- MGA trajectories with multiple DSMs can be modelled in several ways
- Different models can require algorithms with different complexity
- Two different classes of models were developed both based on a linked-conic approximation



Trajectory Models: Velocity Formulation



1. Assign a value to the position vectors of the departure and arrival planets r_{pj} and r_{pj+1} to their corresponding times
2. Assign a value to the times of the DSMs
3. Propagate from the departure time to the time of the first DSMs
4. Assign a value to the three components of the DSM
5. Propagate till the next DSM
6. Continue till the last DSM
7. Compute a Lambert's arc connecting the last DSM with the arrival planet



Trajectory Models: Velocity Formulation

- Each leg requires the previous one in order to propagate
- The position and velocity vectors of the departure and arrival planets can be computed thorugh two variables only: the departure time and the arrival time
- Each DSM requires 4 free variables therefore for n DSMs 4n variables are requried
- Fixing the arrival and departure time, the DSMs can be solved as a 4n dimensional box constrained problem of the kind:

$$\min f_j(\Delta v_1,\ldots,\Delta v_i,\ldots,\Delta v_n,t_1,\ldots,t_i,\ldots,t_n) = \Delta v_{0,j} + \sum_{i=1}^{n} \Delta v_{i,j} + \Delta v_{n+1,j}$$

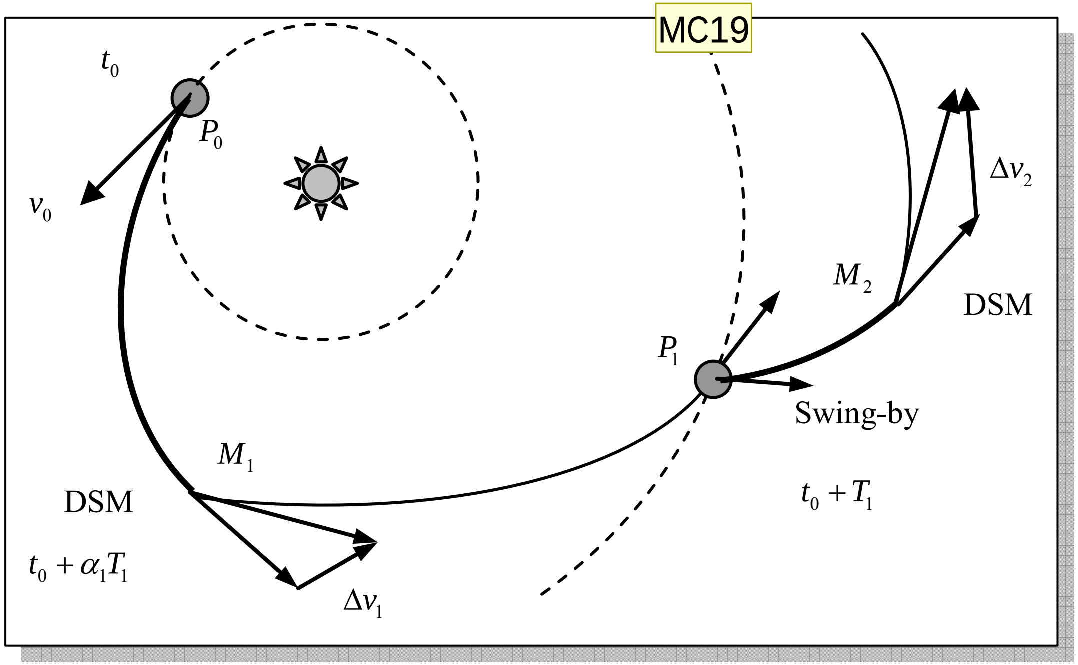
- .

University of Reading

UNIVERSITY of GLASGOW

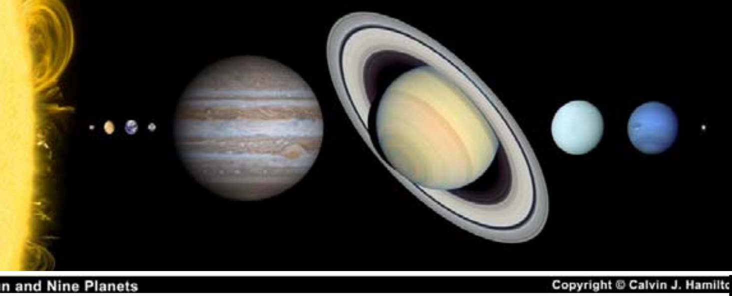
Trajectory Model 1

1. Split the trajectory in phases
2. Each phase connects two celestial bodies
3. Each phase is split into 2 subphases divided by a deep-space manoeuvre
4. At the celestial body the outgoing velocity is computed with the linked conic model
5. The length of each subphase, the departure time and the initial departure velocity must be determined in order to minimise the total Δv



Slide 9

MC19 Aggiornata figura qui sotto
Matteo Ceriotti, 9/6/2007



Gravity Assist Model: Unpowered Swing-by

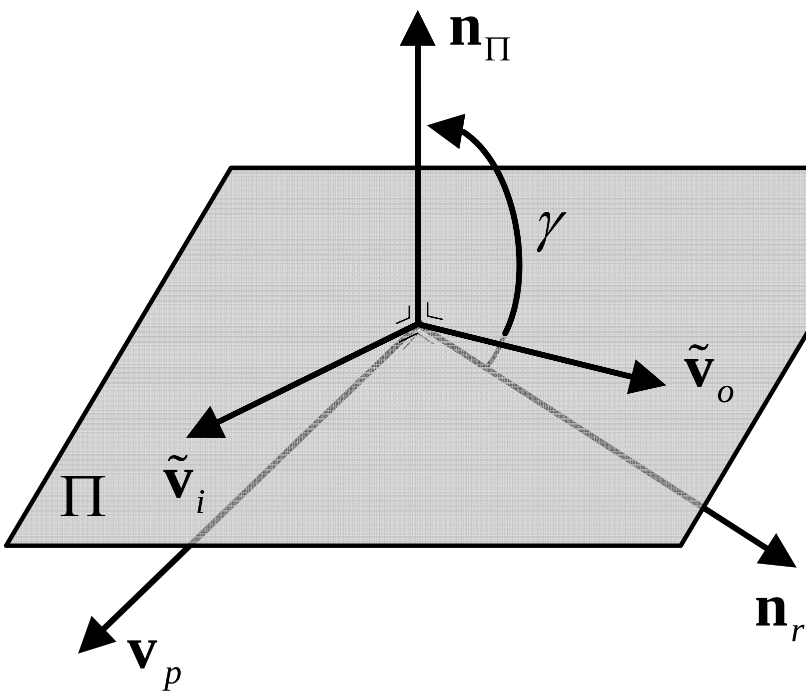
MC20

- Hyperbola plane Π definition

$$\mathbf{n}_r \perp \tilde{\mathbf{v}}_i$$

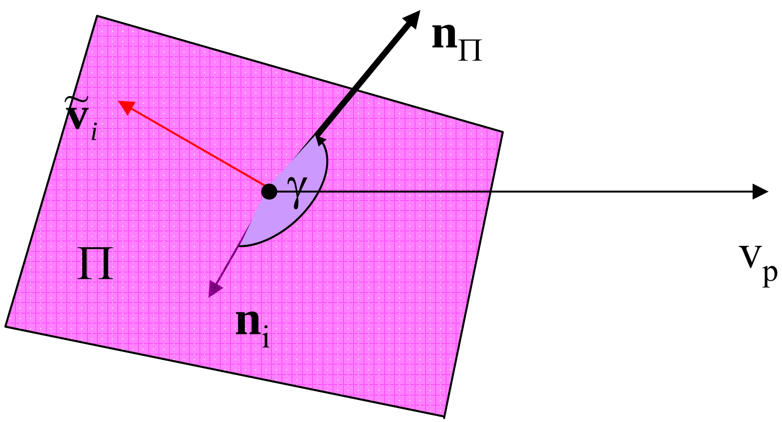
$$\mathbf{n}_r \perp \tilde{\mathbf{v}}_p$$

- Rotation of $\mathbf{n}_\mathrm{r}$ around $\tilde{\mathbf{v}}_i$ of an angle γ



MC20 Propongo questa
Matteo Ceriotti, 9/6/2007

Gravity Assist Model: Unpowered Swing-by



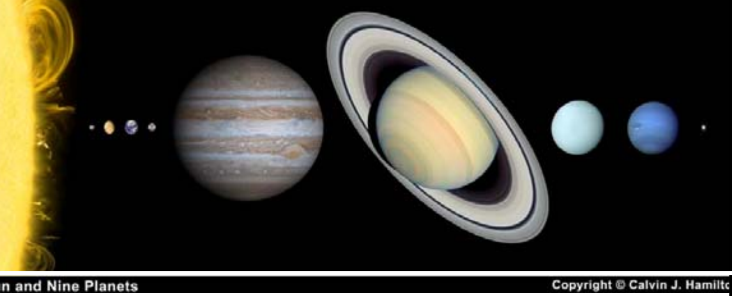
Reference vector $\mathbf{n}_i$ perpendicular to the plane defined by the incoming relative velocity vector and the planet velocity vector

$$\mathbf{u}_i = \frac{\tilde{\mathbf{v}}_i}{\tilde{v}_i}$$

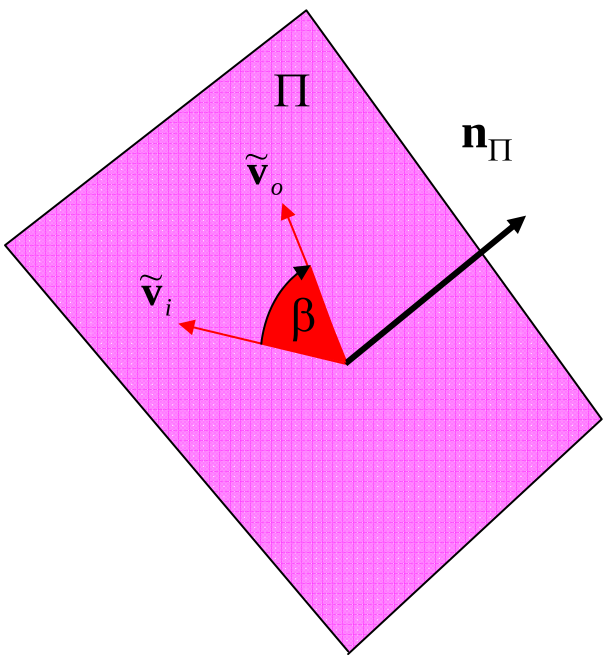
First rotation from $\mathbf{n}_i$ to $\mathbf{n}_\Pi$ the vector normal to the hyperbola plane Π

$$\mathbf{q} = \left[\mathbf{u}_i \sin\frac{\gamma}{2}, \cos\frac{\gamma}{2}\right]^T$$

$$\mathbf{n}_\Pi = Q(\tilde{\mathbf{v}}_i)\mathbf{n}_i$$



Gravity Assist Model: Unpowered Swing-by



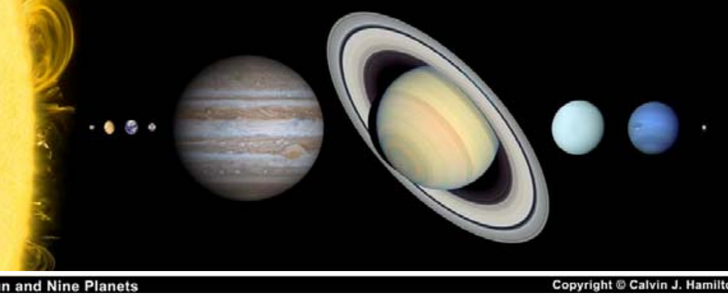
- Second rotation in the plane of the hyperbola, around the vector $\mathbf{n}_\Pi$

$$\mathbf{u}_i = \frac{\tilde{\mathbf{v}}_i}{\tilde{v}_i}$$

- The rotation angle β is a function of the incoming velocity vector and of the pericentre radius r_p

$$\mathbf{q} = \left[\mathbf{n}_\Pi \sin\frac{\beta}{2}, \cos\frac{\beta}{2}\right]^T$$

$$\tilde{\mathbf{v}}_o = Q(\mathbf{n}_\Pi)\tilde{\mathbf{v}}_i$$



Trajectory Model 1

- For a Δv minimisation problem the objective function is the sum of all the Δv's for all the deep space manoeuvres plus the departure Δv:

$$\min_{\mathbf{x}\in D} f(\mathbf{x}) = \sum_{i=0}^{N} \Delta v_i$$

MC21

- The generic solution vector

$$\mathbf{x} = \Big[v_0, \theta, \delta, t_0, T_1, \alpha_1, \gamma_1, r_{p,1}, T_2, \alpha_2, \ldots, \gamma_i, r_{p,i}, T_{i+1}, \alpha_{i+1}, \ldots, \gamma_{N_L-1}, r_{p,N_L-1}, T_{N_L}, \alpha_{N_L} \Big]$$

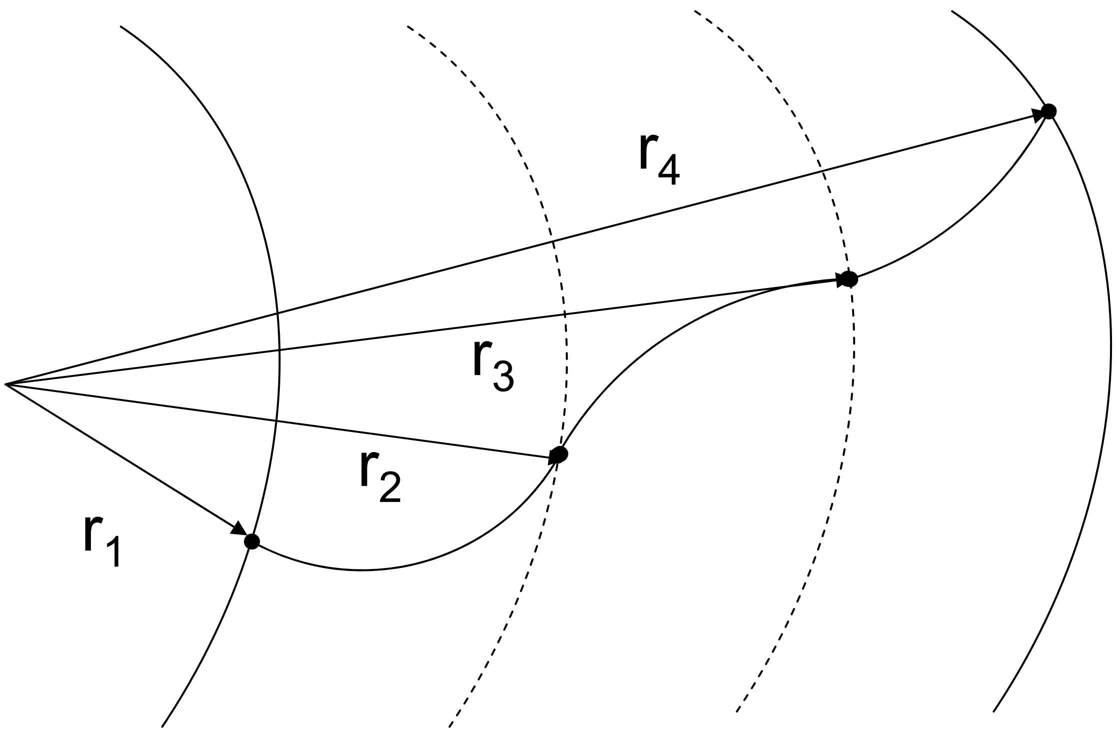
Slide 13

MC21 Sostituito vettore x con quello usato nel paper. Divide le variabili in gruppi di livelli
Matteo Ceriotti, 9/6/2007

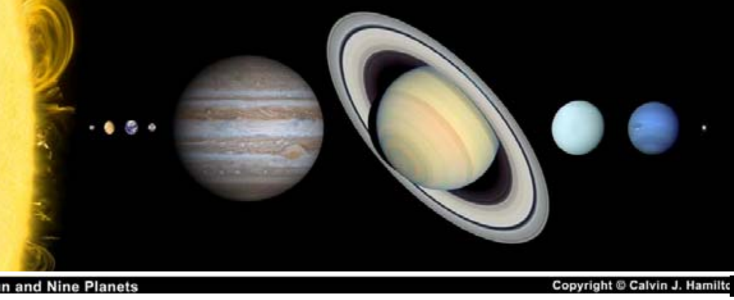
University of Reading

UNIVERSITY of GLASGOW

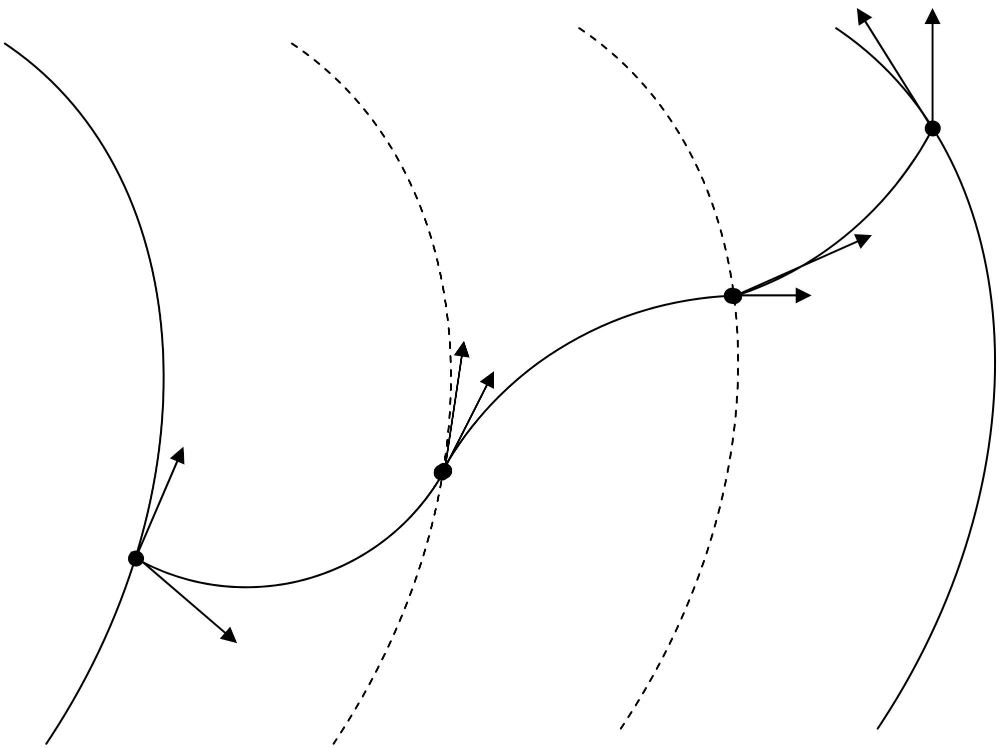
Trajectory Model 2 and the Position Formulation



1. Assign a value to the position vectors of the departure and arrival planets r_j and r_{j+1} and of the DSMs and to their corresponding times
2. Compute the Lambert's arc connecting positions of the planets and of the DSMs

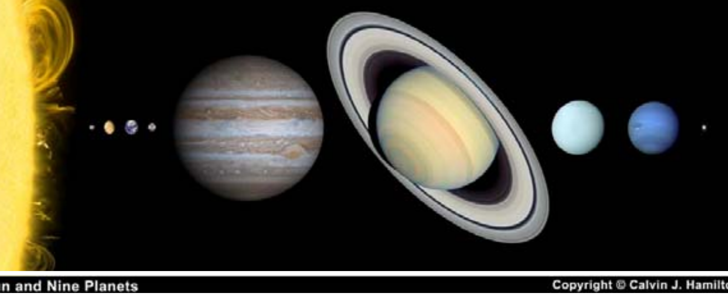


Trajectory Model 2 and the Position Formulation



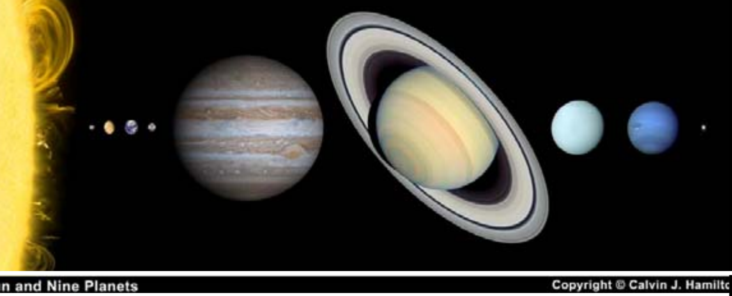
3. Compute the vector difference at each junction point

4. Compute the Δv correction at departure and arrival



Trajectory Models and the Position Formulation

- Each Lambert’s arc connecting two points can be computed independently of the others once the position vectors are assigned.
- The position of the departure and arrival planet can be assigned through two variables only: the departure and the arrival times.
- Each DSM requires four variables instead

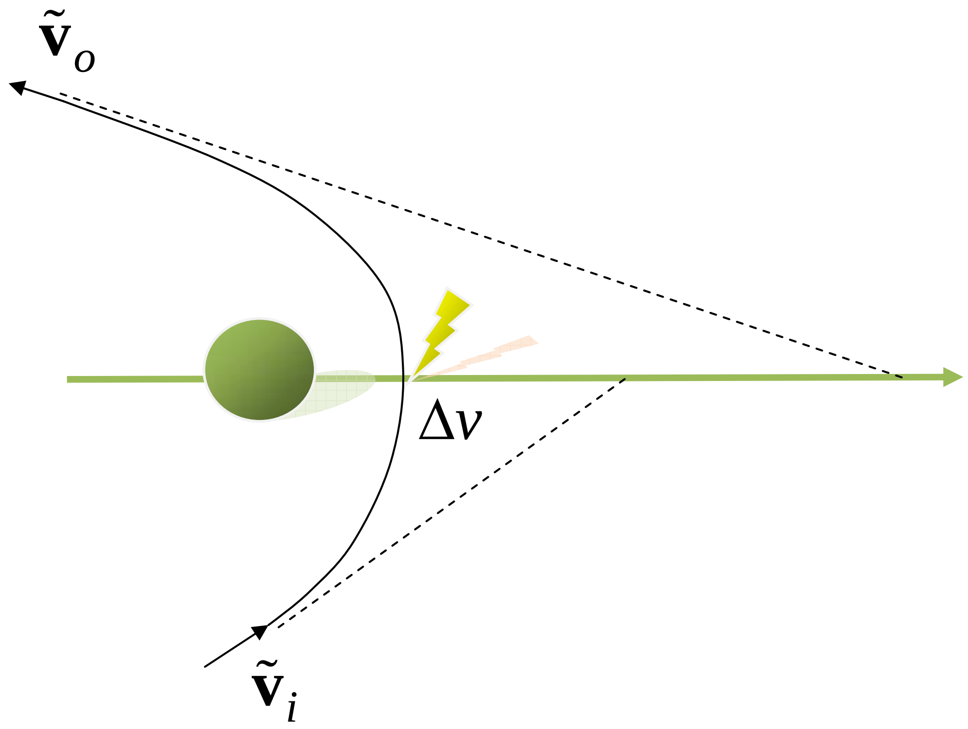


Gravity Assist Model: Powered Swing-by

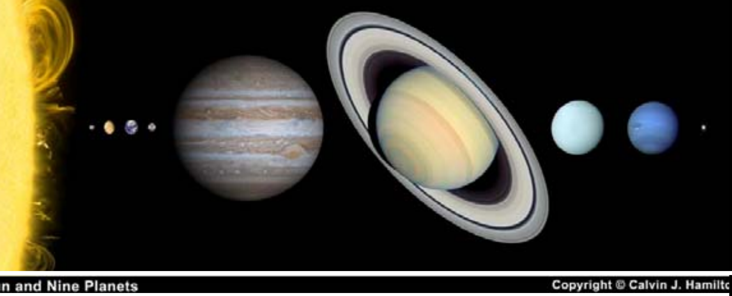
MC15

Given $\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_o$:

- A Newton method is used to find the pericentre radius for the required deviation angle
- If the required deviation can not be achieved computes correction manoeuvre at minimum altitude point

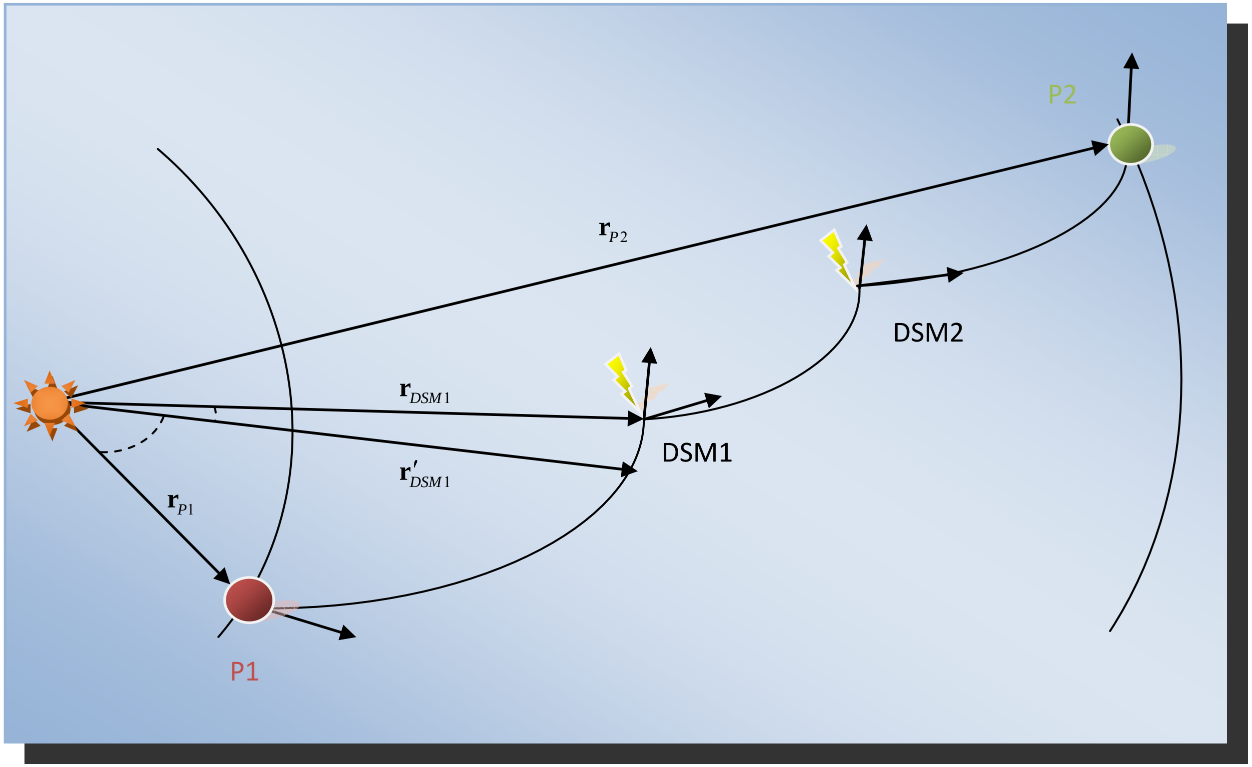


MC15 Inserito modello powered
Matteo Ceriotti, 9/6/2007



Trajectory Model 2

MC16



- Position and velocity of planets from analytical ephemeris.
- Lambert's solution between two DSMs.
- DSM defined by radius, two angles and fraction of the TOF

MC16 Text added

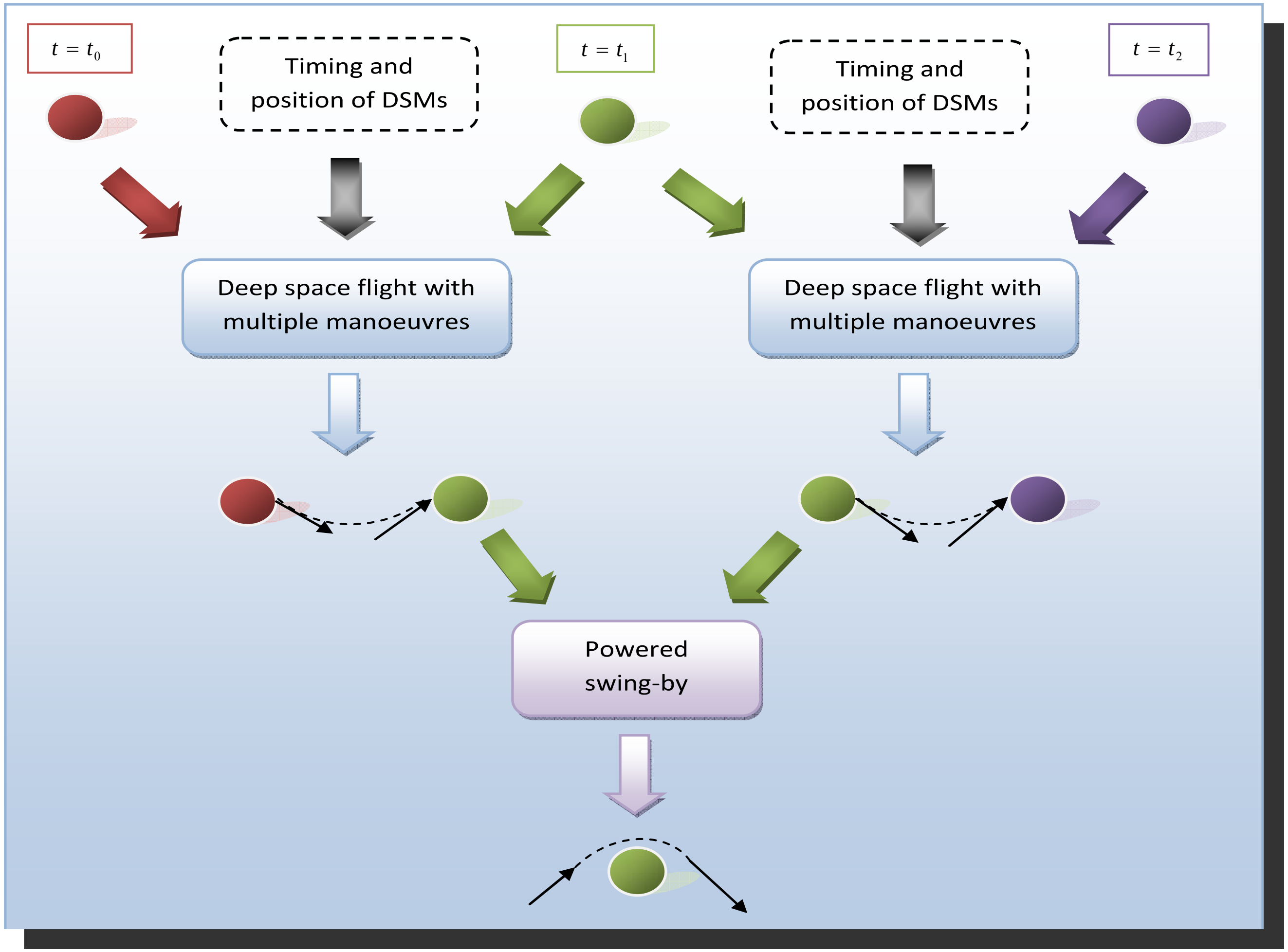
Matteo Ceriotti, 9/6/2007

University of Reading

UNIVERSITY of GLASGOW

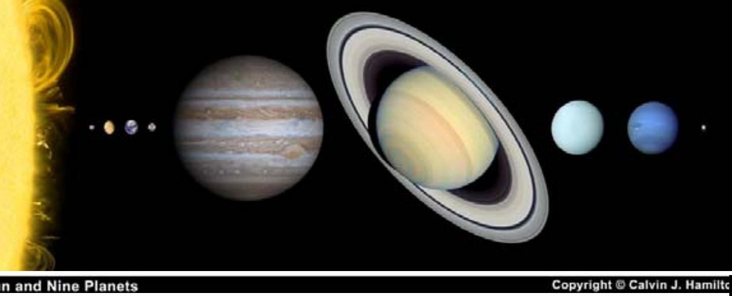
Trajectory Model 2

MC17



MC17 Added text

Matteo Ceriotti, 9/6/2007



University of Reading

UNIVERSITY of GLASGOW

Trajectory Model 2

MC18

- The objective function is the sum of all the Δv's either due to a powered swingby or to a DSM.

$$\min_{\mathbf{x} \in D} f(\mathbf{x}) = \sum_{i=0}^{N} \Delta v_{GA,i} + \sum_{i=0}^{N} \Delta v_{DSM,i} + \Delta v_0$$

- The generic solution vector is:

$$\mathbf{x} = [t_0, r_{1j}, \theta_{1j}, \varphi_{1j}, \alpha_{1j}, T_1, \ldots, r_{ij}, \theta_{ij}, \varphi_{ij}, \alpha_{ij}, T_i, \ldots, r_{Nj}, \theta_{Nj}, \varphi_{Nj}, \alpha_{Nj}, T_N]^T$$

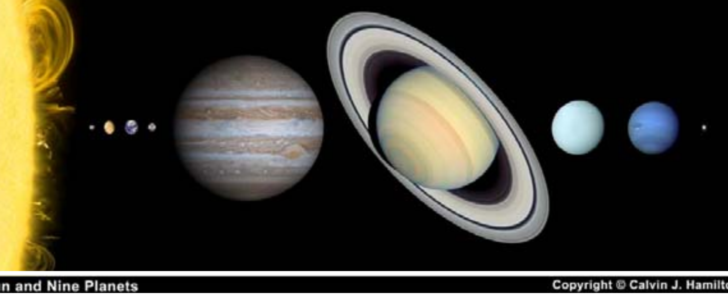
- With the constraints on the pericentre radius of the swingby hyperbolae:

$$\mathbf{r}_p \geq \mathbf{r}_{p_{\min}}$$

Slide 20

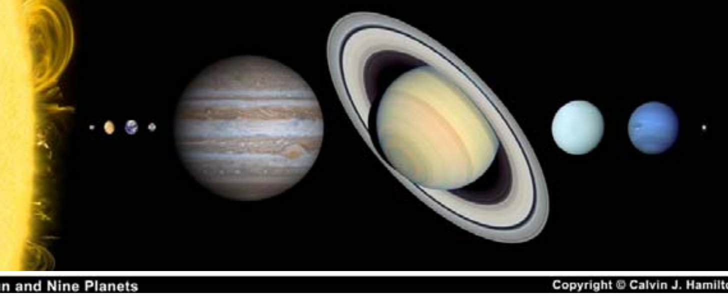
MC18 Added text

Matteo Ceriotti, 9/6/2007



Preliminary Considerations

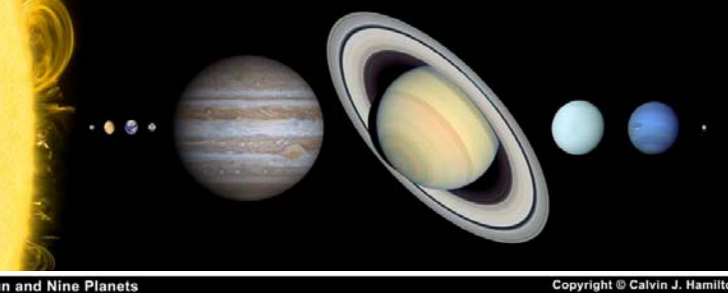
- Model 1 and velocity formulations do not allow to compute each leg independently of the other legs
- Model 2 and position formulations allow the computation of each leg independently of the other legs



Preliminary Considerations

- Model 1 leads to a box constrained problem. Since physical constraint are satisfied intrinsically additional criteria have to be introduced to define the feasibility of the solution.

- Model 2 leads to a constrained optimisation problem with nonlinear constraints. If only feasible solutions are required the problem associated to model 2 reduces to a constraint satisfaction one.

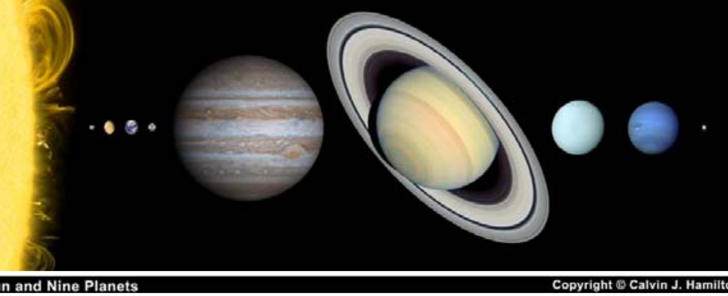


- The two groups in Glasgow and in Reading addressed the problem into two different ways.

- Reading used model 2 while Glasgow used model 1 to build the trajectory.

- Both approaches, however, try to give a positive answer to the reductionist/holistic question:

Is the sum of the parts equivalent to the whole?



Some preliminary Answers

- The sum of the global optima for each subproblem in which we can decompose model 1 does not always correspond to the global optimum for the sum of the subproblems.

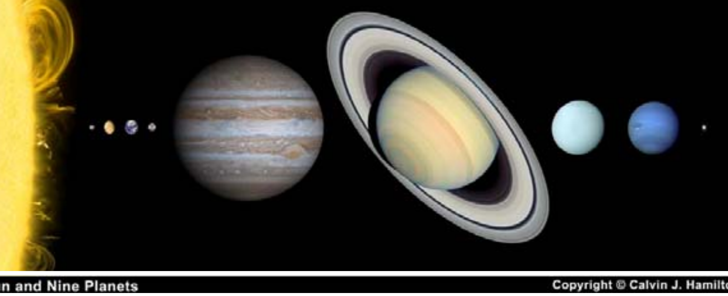
- The choice of the function f_k associated to each subproblem can change the search for the global solution.

- If we define a box contained in the subspace D_k as:

$$X_{k,q} \subset D_k$$

- Then we can find a union of $Q(k)$ boxes for each slice k the cartesian product of which contains the global optimum of the whole problem:

$$\mathbf{x}_{global} \in \prod_{k=1}^{M} \bigcup_{q=1}^{Q(k)} X_{k,q}$$

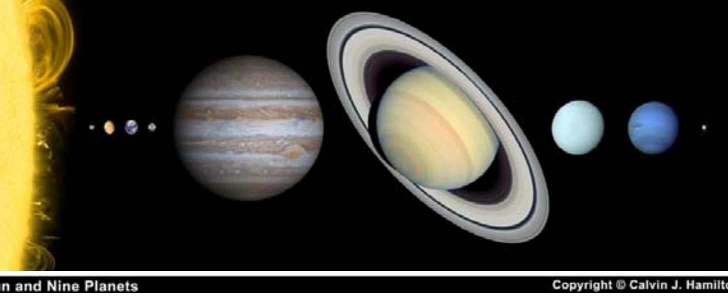


University of Reading

UNIVERSITY of GLASGOW

Searching Through an Optimiser

- Once the problem is **properly** decomposed each subproblem can be addressed separately if a set of **appropriate criteria** can be found that makes the sum of the parts equivalent to the whole
- Each single leg can be optimised either locally or globally with respect to one or more criteria or…
- …for each single leg an optimiser can be used to look just for a set of feasible solutions



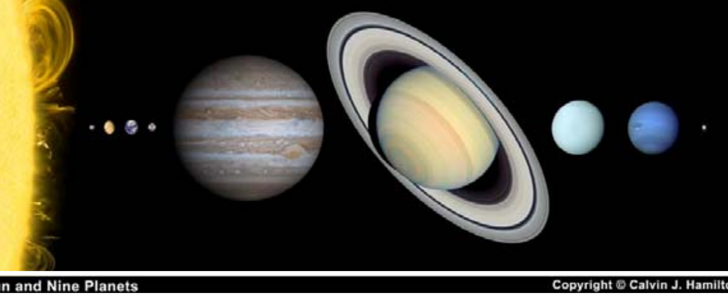
University of Reading

UNIVERSITY of GLASGOW

Local Optimisation

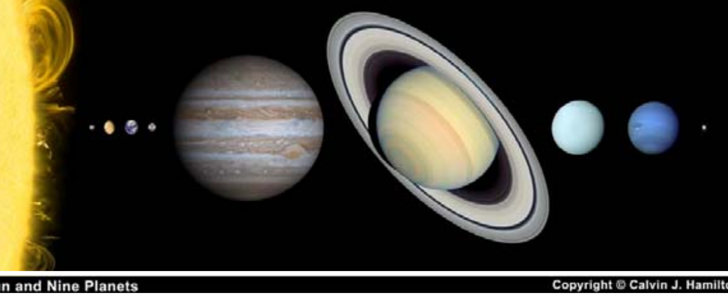
- From a set of preliminary tests it was observed that for each subproblem a set of multiple minima exists. This holds true even if the departure date and arrival dates for each leg are fixed. The result is consistent with the use of a Lambert's solver (or equivalent).

- Optimising locally each subproblem can result into a suboptimal solution for the whole trajectory, i.e. the sum of the parts is not equivalent to the whole



University of Reading

- Pruning Algorithm
- Test Cases: Messenger, Bepi Colombo missions
- Sequence Optimizer
- Test Cases including 1st ACT Trajectory Competition



MSOP: Find

$$\mathbf{x} = [\ \mathbf{x}_1^T,\ \ \mathbf{x}_1^T,\ \ \cdots,\ \ \mathbf{x}_{s+1}^T\ ]^T \in \Omega$$

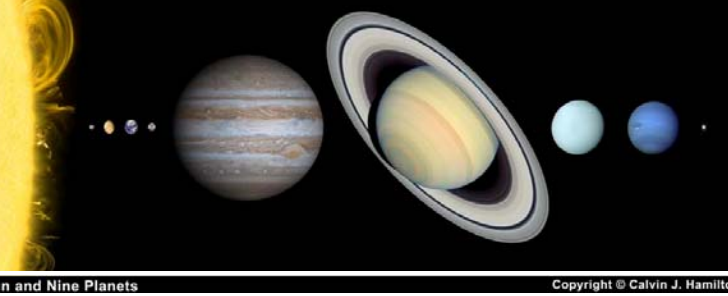
to minimise

$$f(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{s+1})$$

subject to

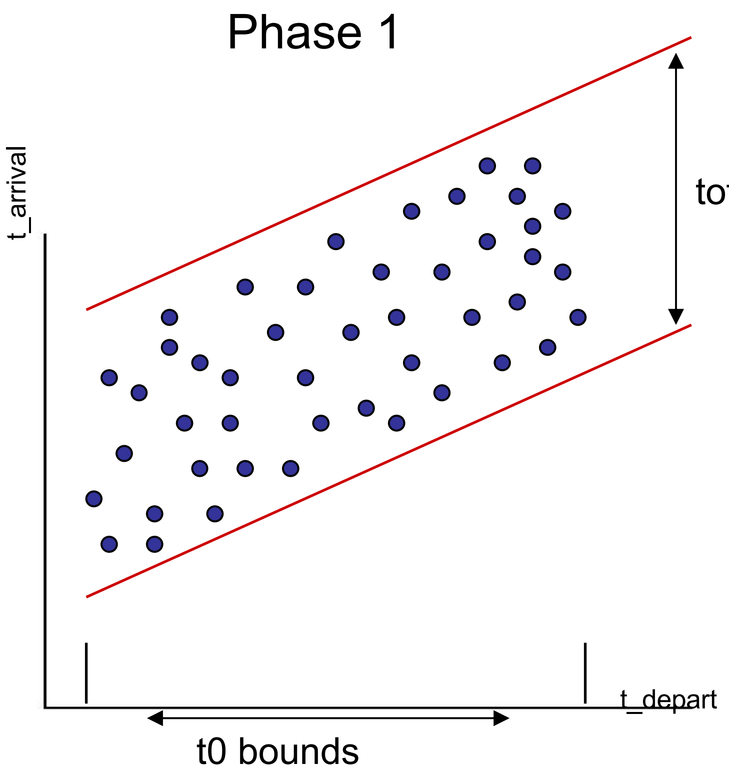
$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_1, \ldots, \mathbf{x}_{k+1}),\ k = 1, \ldots, s$$

$$\mathbf{g}_k(\mathbf{z}_k) \leq 0,\ k = 1, \ldots, s+1$$

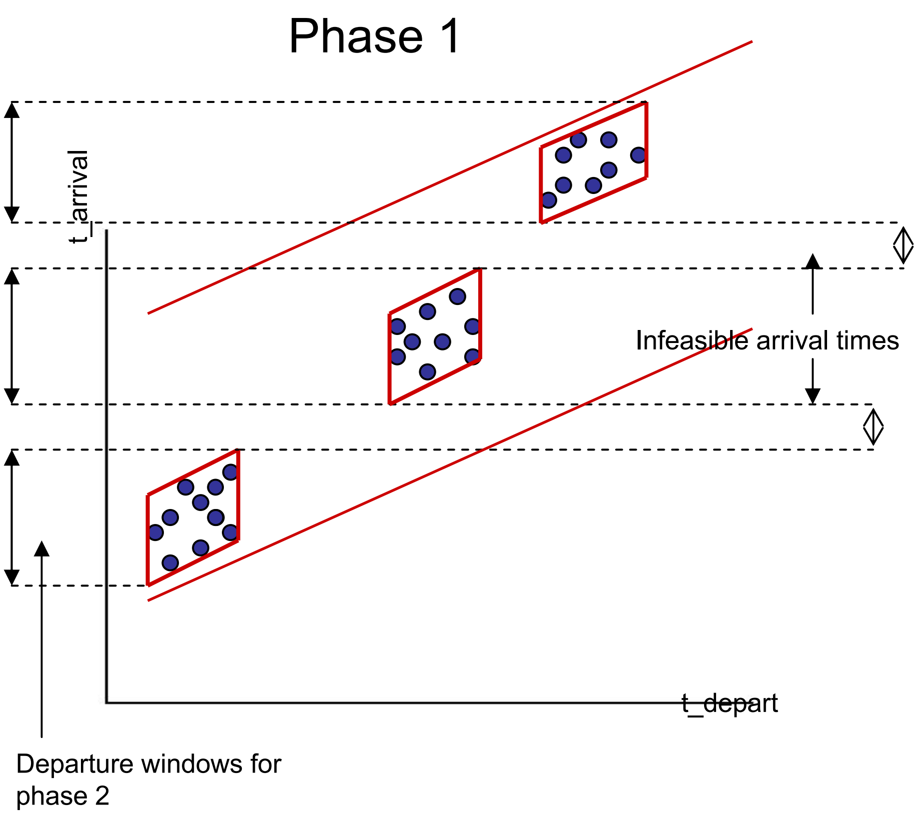


GASP Algorithm with DSM

- Inspired by original GASP ideas
- Sampling based on local optimization
- Clustering employed to estimate feasible regions
- Description based on two phases, but easily extendible to more phases.
- One DSM per phase considered, but more can be added.
- Initial search space is a hyperrectangle (bounds)



1) Initialise the local optimiser to N starting points.



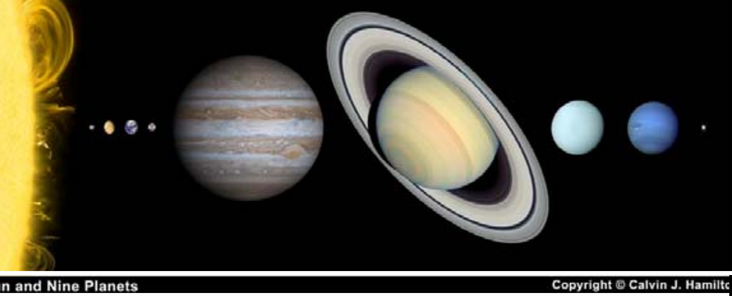
1) Initialise the local optimiser to N starting points.

2) Run clustering algorithm to group remaining feasible points.

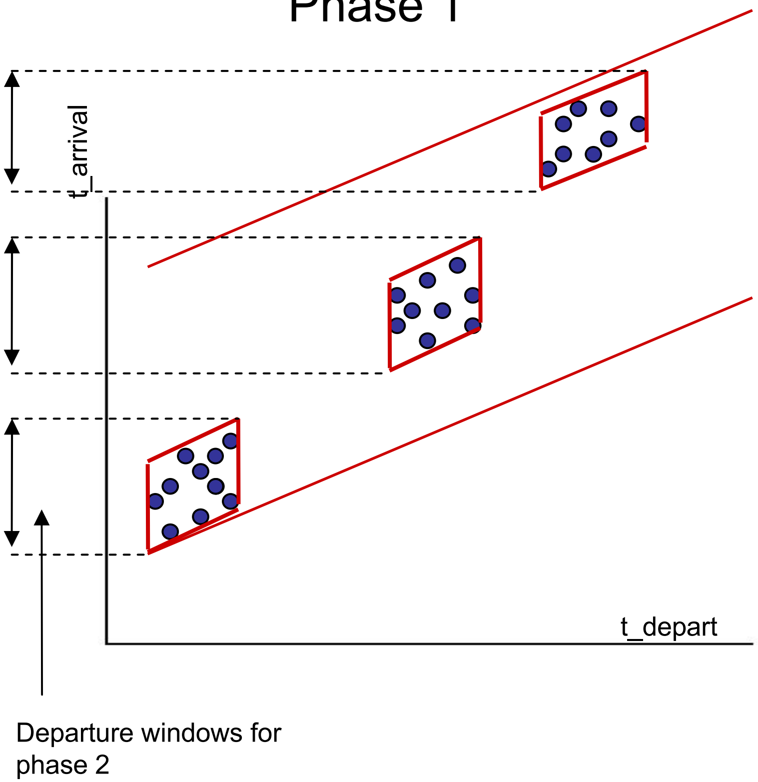
3) Generate bounding boxes from the extreme points in each cluster

University of Reading

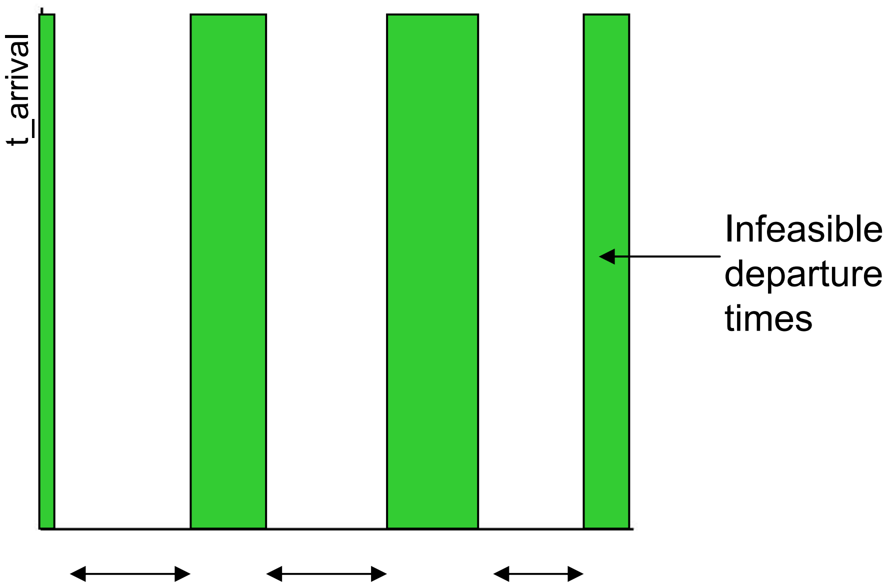
UNIVERSITY of GLASGOW



Phase 1



Phase 2



Optimization Problem_1

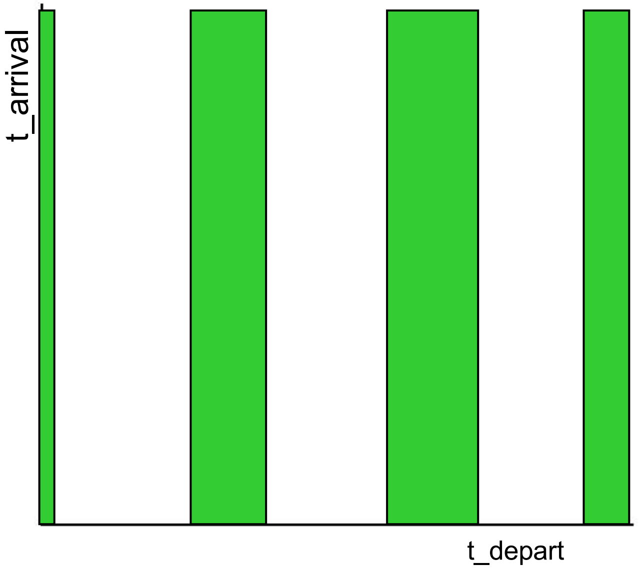
Optimization Problem_2

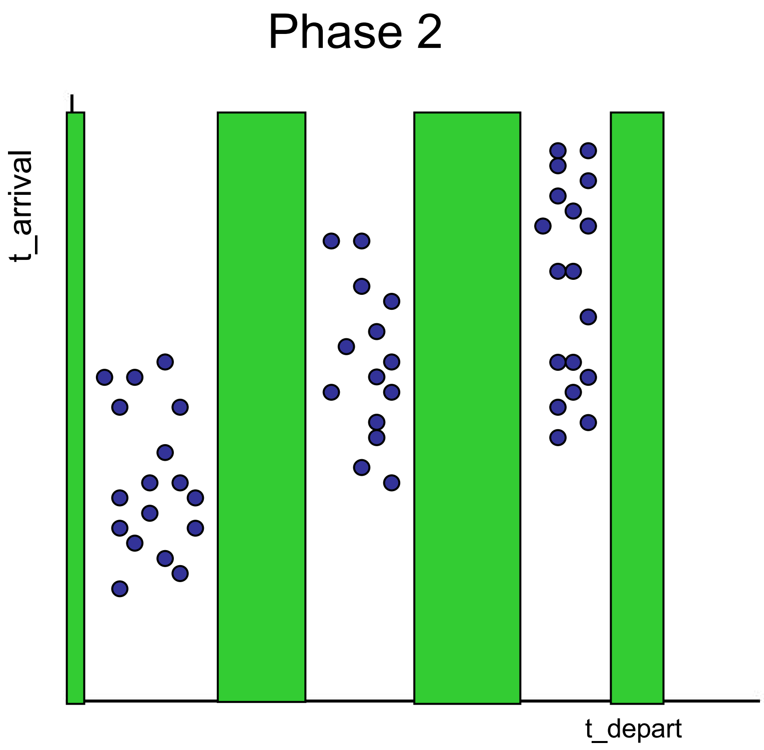
University of Reading

UNIVERSITY of GLASGOW

Phase 2

Phase 2



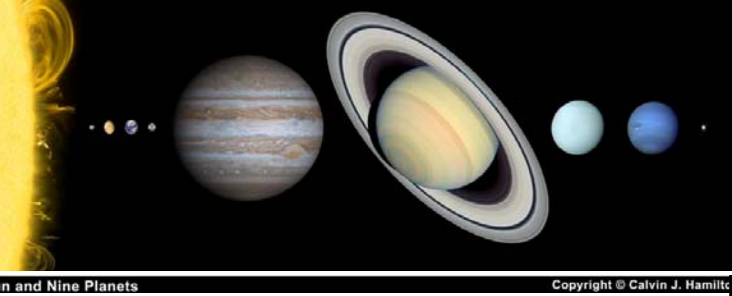


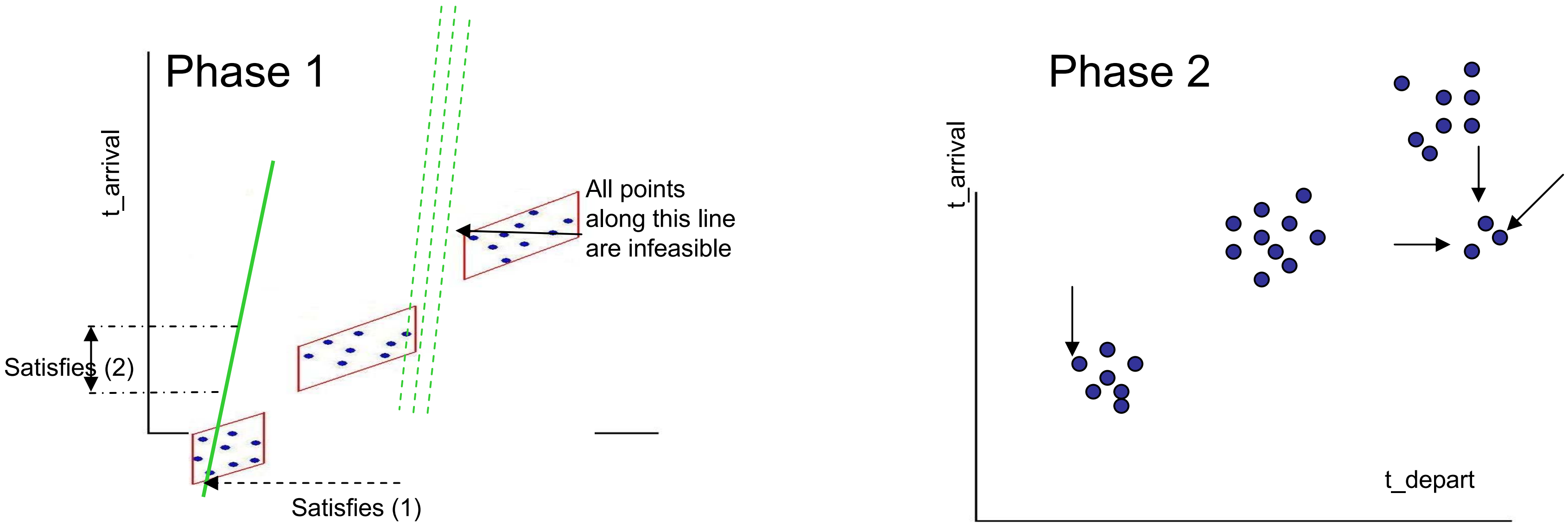
1) Initialise the local optimizer n times in each window

2) Optimize each point in turn

Optimization Problem_2

Optimization Problem_1



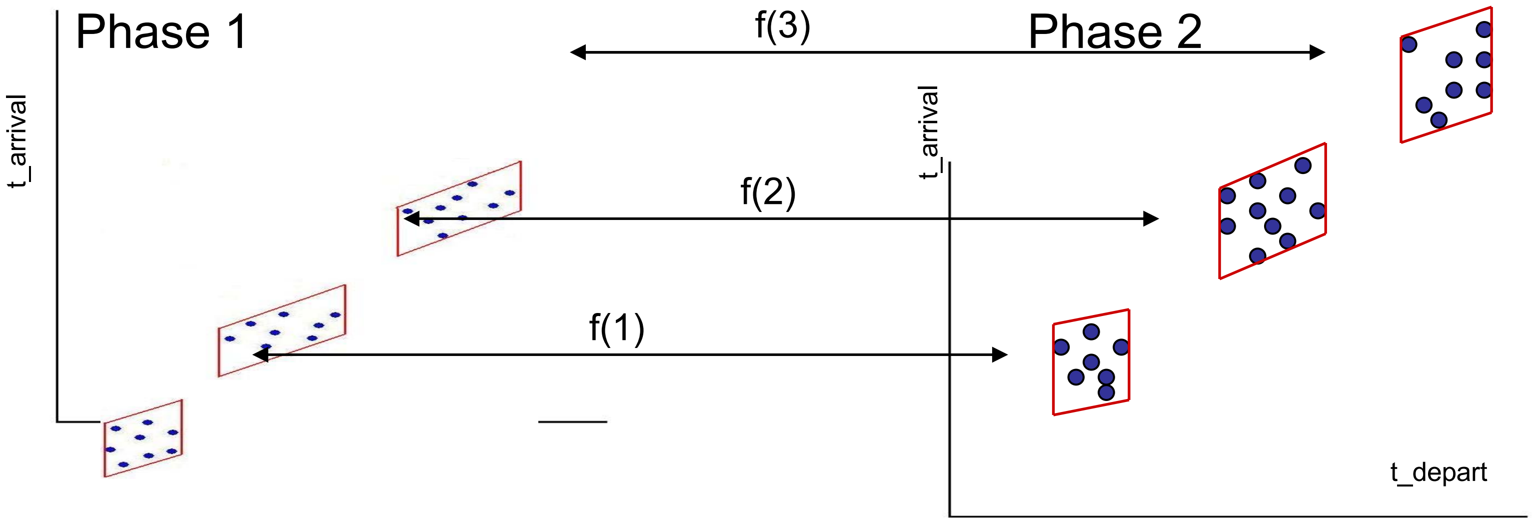


3) Each point in phase 2 is associated with a departure velocity. The feasibility of a point in phase 2 w.r.t the preceding swingby manoeuvre is checked by solving a local optimization at phase one.

4) If after *k* attempts no feasible point is found in phase 1, discard the point in phase 2.

University of Reading

UNIVERSITY of GLASGOW



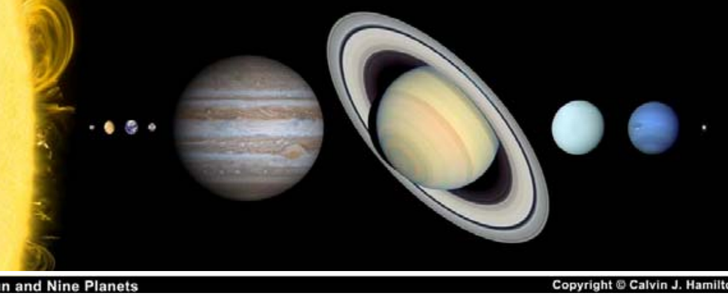
5) Run clustering algorithm on remaining points in phase 2, generate bounding boxes

6) Backward constraining: re-cluster phase 1 with new feasible points corresponding to points in phase 2

7) Locate solution families

8) Explore the solution families by globally optimizing each family for a limited number of generations

9) Further explore the family that produces the lowest objective function value in step 8



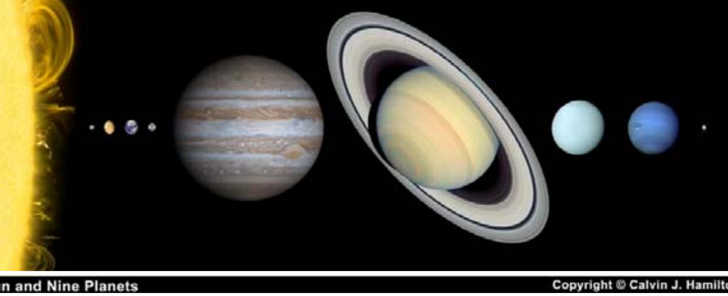
Pruning Algorithm Complexity

Total number of Lambert problem calls:

$$l = s_1(d_1+1)k_1 + k_2 \sum_{i=2}^{n} \{s_i w_i (d_i+1)\} + k_3 \sum_{i=2}^{n-1} \{s_i w_i (d_i+1)\}$$

l	calls to Lambert solver
n	number of phases
s_1	number of local optimisations in first phase
s_i	number of local optimisations per window $(i > 1)$
d_i	number of DSM's in phase i
w_i	departure windows found in phase i
k_i	average number of function evaluations in phase i

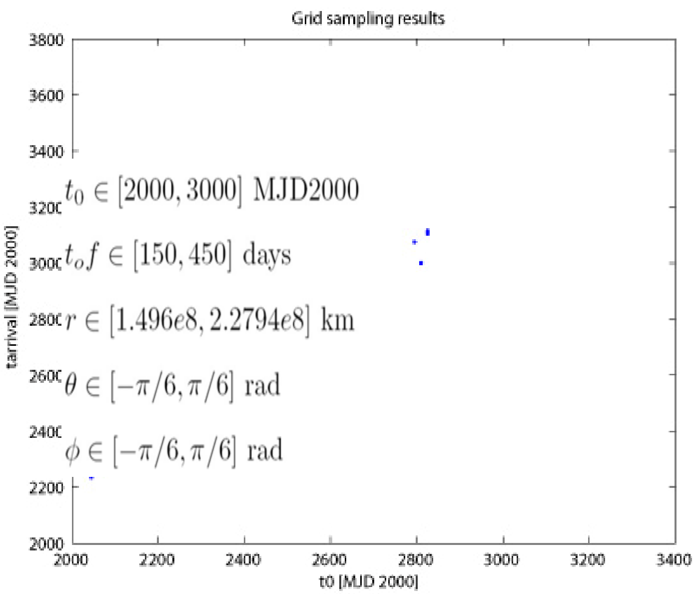
It is possible to bound the the number of Lambert calls with a linear function of the number of phases



University of Reading

UNIVERSITY of GLASGOW

ans



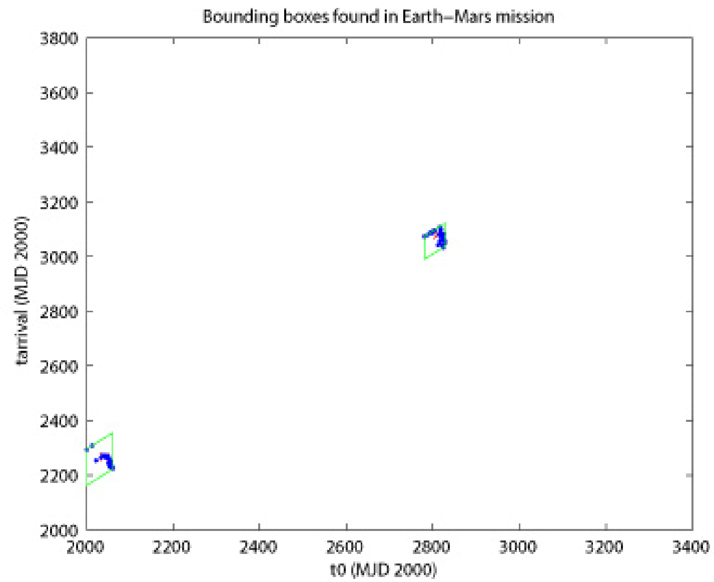
$t_0 \in [2000, 3000]$ MJD2000

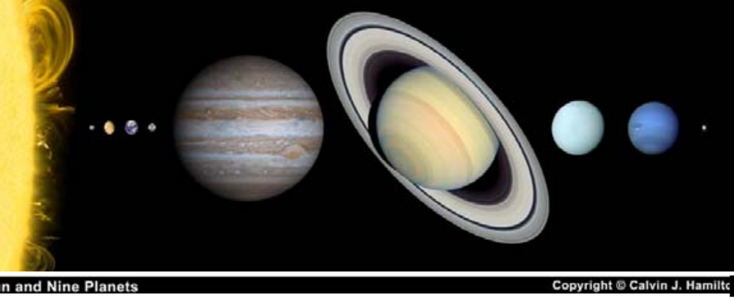
$t_o f \in [150, 450]$ days

$r \in [1.496e8, 2.2794e8]$ km

$\theta \in [-\pi/6, \pi/6]$ rad

$\phi \in [-\pi/6, \pi/6]$ rad





University of Reading

UNIVERSITY of GLASGOW

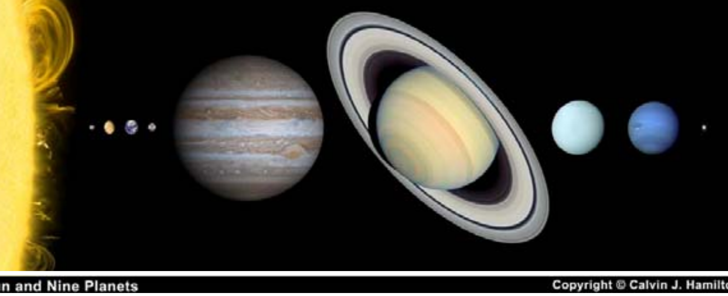
Comparison:

Grid Sampling

- Sampling resolution of 45*20*20*10*10*10
  (18,000,000 points)
- Constraint values had to be doubled to find
  feasible points
- 3.36 hours to run on 2GHz PC
- 36 million calls to Lambert Solver
- Only 19 feasible points found

Pruning Algorithm

- 150 random 6D vectors generated
- 128 s to run on 2GHz PC
- 116,326 calls to the Lambert solver
- 89 feasible points found



Example: E-V-M

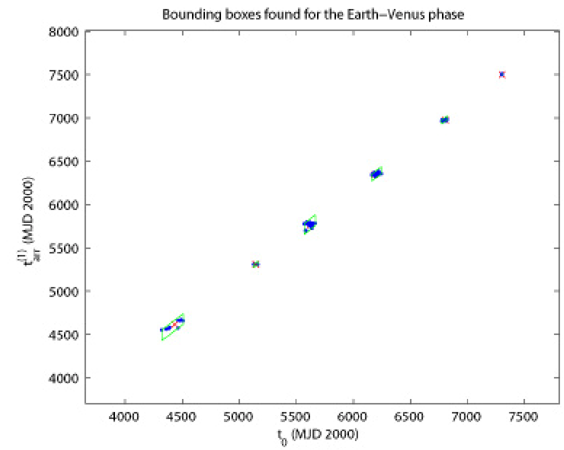
- No DSM in phase 1
- 1 DSM in phase 2
- Launch, swingby, DSM
- Insertio rp e
- Bounds:

$$t_0 \in [36507302] \text{ MJD2000}$$
$$t_{of}^{(1)} \in [50, 400]$$
$$t_{of}^{(2)} \in [50, 700]$$
$$r \in [1.0821e8, 2.2794e8]$$
$$\theta \in [-\pi, \pi]$$
$$\phi \in [-\pi/8, \pi/8]$$
$$\alpha \in [0.1, 0.9]$$

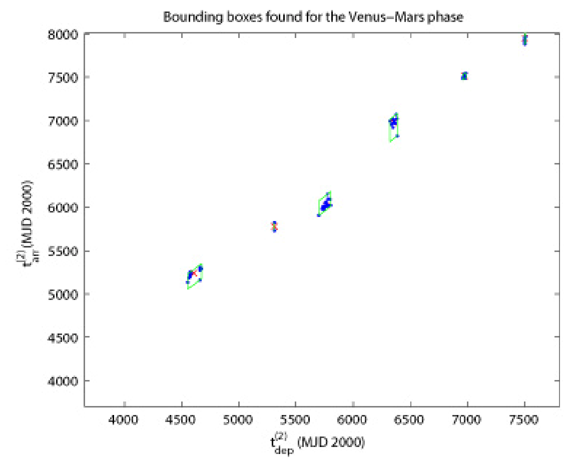
- Constraint:

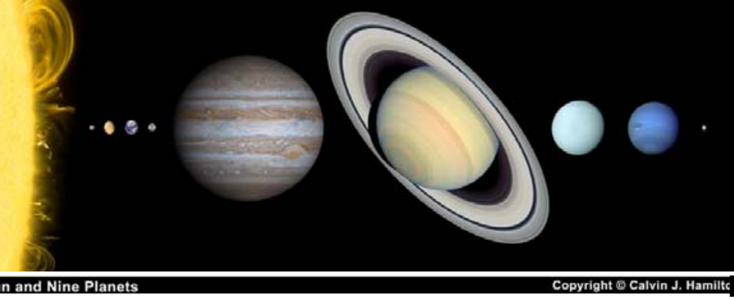
$$\Delta v_{\text{dep}} \leq 5\,\text{km/s}$$
$$\Delta v_{\text{ga}} \leq 5\,\text{km/s}$$
$$\Delta v_{\text{DSM}} \leq 2\,\text{km/s}$$
$$\Delta v_{\text{b}} \leq 3\,\text{km/s}$$

Phase 1

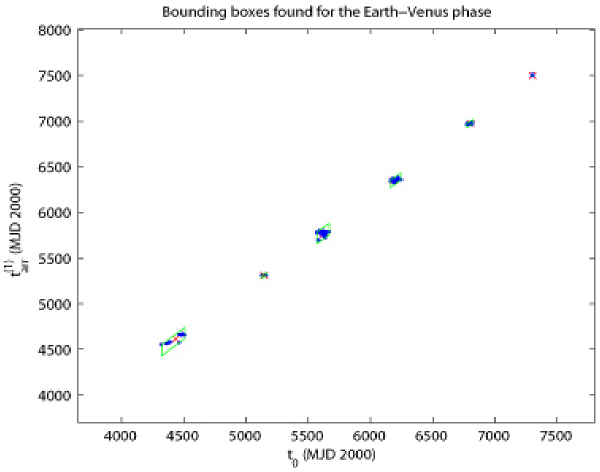


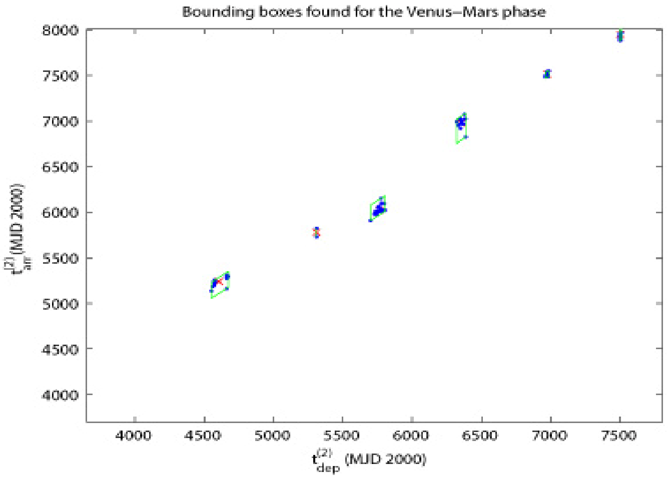
Phase 2



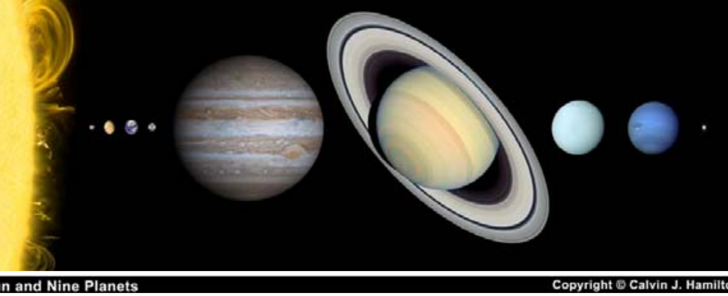


E-V-M with 1 DSM





- Majority of the search space is unfeasible
- Works well over large launch window
- Combines phases with purely ballistic trajectories with more complex cases
- Phase 1 < 30s
- Phase 2 < 2.5mins
- 6 solution families found



University of Reading

UNIVERSITY of GLASGOW

Global Optimization

1. Apply differential evolution to each of the 6 solution families. Allow a maximum of 300 generations with a population size of 20
2. Look at the objective function value of each family, select family with best solution
3. Re-optimize best family, this time allowing for 2000 generations

$$\min_{\mathbf{x}} f(\mathbf{x}) = \Delta v_{\text{dep}}^{(1)} + \Delta v_{\text{ga}}^{(1)} + \Delta v_{\text{DSM}}^{(2)} + \Delta v_{\text{b}}^{(2)}$$

Solution

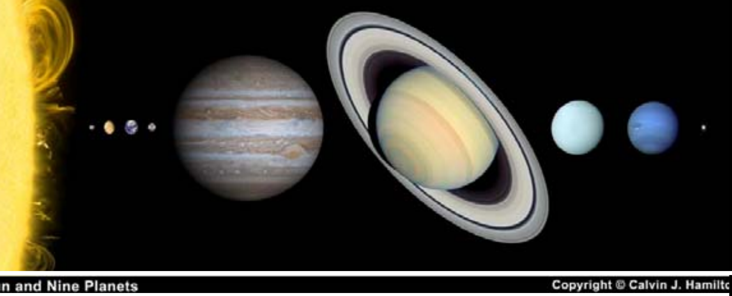
$t_0 = 4469.9$
$t_{\text{of}}^{(1)} = 171.7$
$t_{\text{of}}^{(2)} = 682.4$
$r^{(2)} = 1.725$
$\theta^{(2)} = -1.91$
$\phi^{(2)} = -0.0$
$\alpha^{(2)} = 0.503$

subject to $\mathbf{x} \in B_s$

$$\begin{bmatrix} \Delta v_{\text{dep}}^{(1)} \\ \Delta v_{\text{DSM}}^{(1)} \\ \mathbf{v}_{\text{in}}^{(1)} \\ \mathbf{v}_{\text{out}}^{(2)} \\ \Delta v_{\text{DSM}}^{(2)} \end{bmatrix} = \mathbf{h}(\mathbf{x})$$

$$\Delta v_{\text{ga}}^{(1)} = q_1(\mathbf{v}_{\text{in}}^{(1)}, \mathbf{v}_{\text{out}}^{(2,i)}, r_{\text{min}}^{(1)})$$

ce
re
x 10^8



Case study 1: Bepi-Colombo Mission

Reduced trajectory: E-E-V-V-Me
21-dimensional problem
8 Delta-V's

Constraints

$$\Delta v_{\mathrm{dep}} \leq 3\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{ga}}^{(1..3)} \leq 3\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{DSM}}^{(1..4)} \leq 2\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{b}} \leq 3\,\mathrm{km/s}$$

Bounds

$$t_0 \in [4000, 8000]$$

$$tof^{(1..4)} \in [100, 800]$$

$$r^{(1)} \in [1.3464e8, 1.6456e8]$$

$$r^{(2)} \in [1.0821e8, 1.4960e8]$$

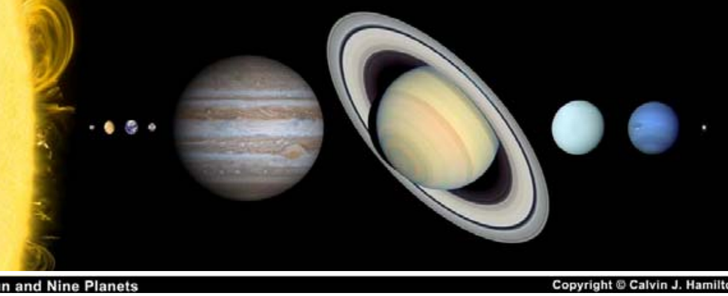
$$r^{(3)} \in [9.7388e7, 1.1903e8]$$

$$r^{(4)} \in [5.7909e7, 1.0821e8]$$

$$\theta^{(1..4)} \in [-\pi, \pi]$$

$$\phi^{(1..4)} \in [-\pi/6, \pi/6]$$

$$\alpha^{(1..4)} \in [0.1, 0.9]$$

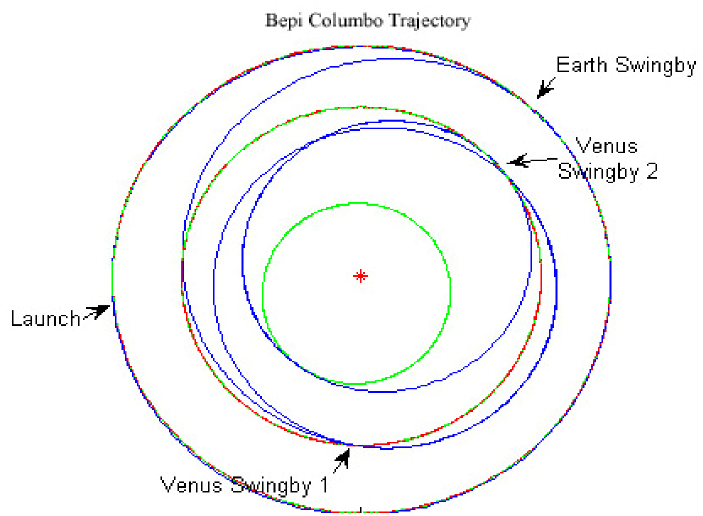


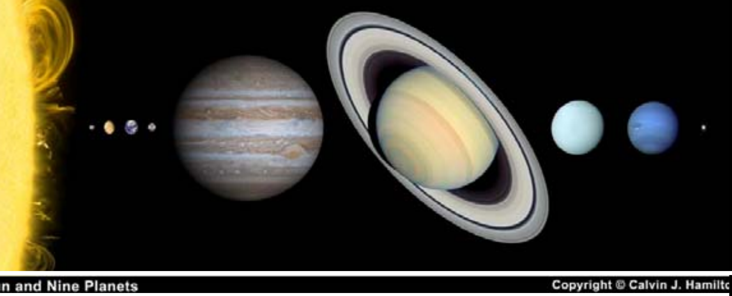
Case study 1: Bepi Colombo Mission

Results

index	t (MJD2000)	tof (days)	r (km)	θ rad	ϕ rad	α
0	7.5983e+003					
1		372.3406	1.5203e+008	-2.8955	0.0001	0.5166
2		171.2483	1.2460e+008	1.4827	0.0143	0.4886
3		305.9244	9.7390e+007	-0.6870	-0.0000	0.6110
4		256.0651	9.7822e+007	-0.8699	-0.0730	0.5178

Manoeuvre	Type	Bodies Involved	$\Delta V (km/s)$	Constraint Violation
1	Launch	E	0.274406	No
2	DSM	E-E	0.263321	No
3	DSM	E-V	0.010990	No
4	DSM	V-V	0.036779	No
5	DSM	V-Me	0.188297	No
9	Swingby	E	0.413114	No
10	Swingby	V	0.002458	No
11	Swingby	V	0.006241	No
		Total ΔV	1.195605	





Case study 2: Messenger Mission

Sequence: E-E-V-V-Me-Me-Me

Challenge

- 7 phase mission
- 1 DSM per phase
- 6 swingbys
- Insertion manoeuvre
- 36-dimensional search space
- 15 constraints

Constraints

$$\Delta v_{\mathrm{dep}} \leq 5\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{ga}}^{(1..6)} \leq 2\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{DSM}}^{(1..7)} \leq 3\,\mathrm{km/s}$$

$$\Delta v_{\mathrm{b}} \leq 4\,\mathrm{km/s}$$

Bounds

$$t_0 \in [1000, 4000]$$

$$tof^{(1..7)} \in [200, 500]$$

$$r^{(1)} \in [1.3464e8, 1.6456e8]$$

$$r^{(2)} \in [1.0821e8, 1.4960e8]$$

$$r^{(3)} \in [9.739e7, 1.1903e8]$$

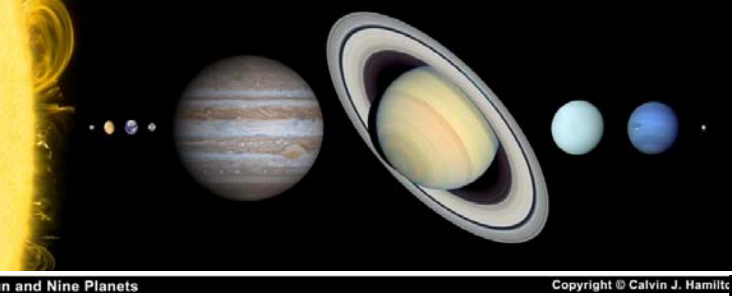
$$r^{(4)} \in [5.791e7, 1.0821e8]$$

$$r^{(6..7)} \in [5.212e7, 6.370e7]$$

$$\theta^{(1..7)} \in [-\pi, \pi]$$

$$\phi^{(1..7)} \in [-\pi/6, \pi/6]$$

$$\alpha^{(1..7)} \in [0.1, 0.9]$$



Case study 2: Messenger Mission

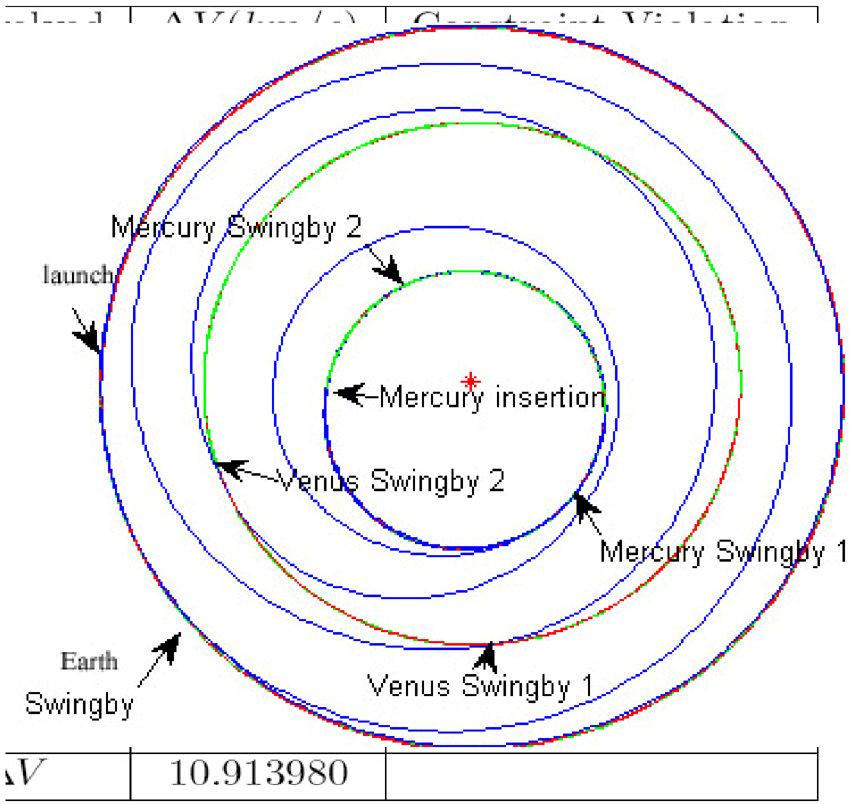
Results:

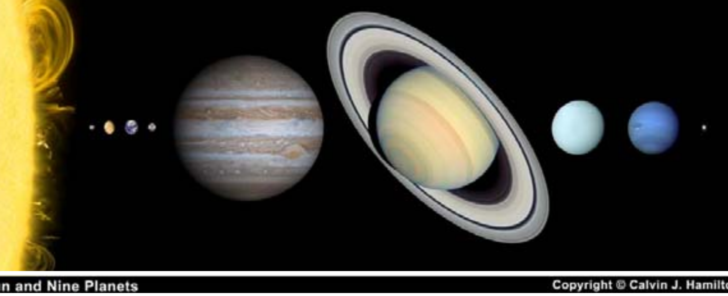
index	t (MJD2000)	tof (days)	r (km)	θ rad	ϕ rad	α
0	1527.3					
1		399.9972	1.4993e+8	-3.1117	6.0655e-6	0.4658
2		377.1584	1.3667e+8	-1.4910	0.0176	0.6430
3		177.7119	1.0398e+8	2.2906	6.7708e-4	0.3966
4		171.7616	5.8483e+7	-2.8456	0.0197	0.4707
5		126.5759	6.7573e+7	-1.1223	3.6121e-4	0.5166
6		132.5413	5.2811e+7	-2.2575	-0.0053	0.4979
7		107.1364	6.7171e+7	-1.0448	-6.0022e-5	0.6219

Case study 2: Messenger Mission

Results:

Manoeuvre	Type	Bodies Involved	$\Delta V(km/s)$	Constraint Violation
1	Launch	E	0.127137	No
2	DSM	E-E	0.074635	No
3	DSM	E-V	2.253777	No
4	DSM	V-V	1.621058	No
5	DSM	V-Me	3.085997	Yes
6	DSM	Me-Me	0.094500	No
7	DSM	Me-Me	0.166976	No
8	DSM	Me-Me	0.063729	No
9	Swingby	E	0.509242	No
10	Swingby	V	0.005469	No
11	Swingby	V	0.843764	No
12	Swingby	Me	1.774886	No
13	Swingby	Me	0.015440	No
14	Swingby	Me	0.001736	No
15	Insertion	V	0.275635	No
		Total ΔV	10.913980	





Sequence Optimization

Top level: integer optimization

Number of phases

Swingby sequence (which planets to visit)

Number of DSM in each phase

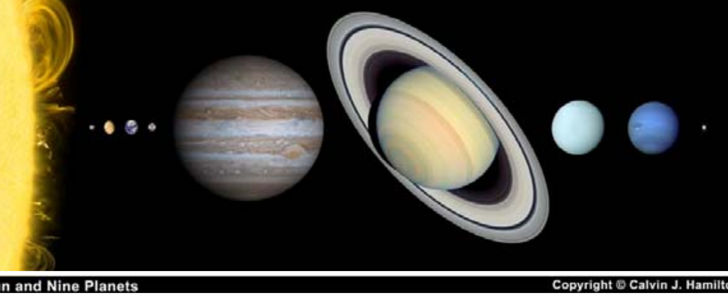
↓ ↑

Lower level:

Pruning + continuous optimization

Timing parameters

DSM parameters



Top Level Optimization

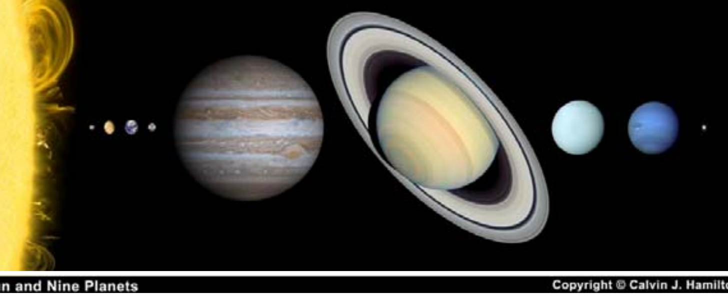
Non-linear Integer Programming

Strategy: (from *"PSO for Integer Programming"* Laskari et al)

Using a standard global optimizer

1) Evaluate objective function in usual manner
2) Once the optimizer finishes round the values in the solution vector to the nearest whole number

Problem: which global optimizer to use?



University of Reading

UNIVERSITY of GLASGOW

Differential Evolution vs. PSO

- Selection of problems with known optimal minima chosen
- Try to get a feel for how each optimizer performs

Problems included: (but not limited to)

$$J_1(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$$

$$x^* = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$$

$$J_1(x^*) = 0$$

$$J_3(x) = -\begin{bmatrix} 15 & 27 & 36 & 18 & 12 \end{bmatrix} x + x^T \begin{bmatrix} 35 & -20 & -10 & 32 & -10 \\ -20 & 40 & -6 & -31 & 32 \\ -10 & -6 & 11 & -6 & -10 \\ 32 & -31 & -6 & 38 & -20 \\ -10 & 32 & -10 & -20 & 31 \end{bmatrix} x$$

$$x^* = \begin{bmatrix} 0 & 12 & 23 & 17 & 6 \end{bmatrix}^T$$

$$J_3(x^*) = -737$$

Results

- PSO outperformed DE in every test
- After 100 trials PSO converged to the optimal solution 100% of the time
- DE converged to the optimal solution 78% of the time
- PSO required fewer generations to converge to optimal solution on every occasion

Problem Formulation:

Departure planet: P_{dep}

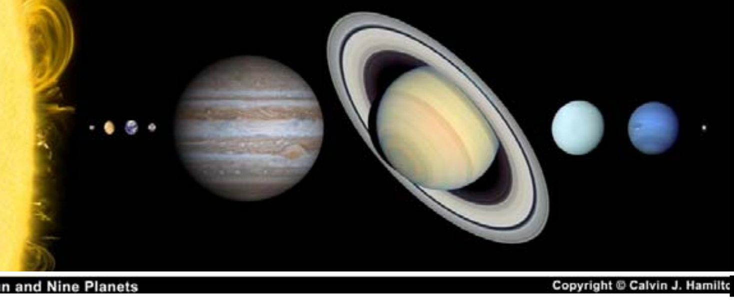
Destination planet: P_{dest}

Maximum number of phases: n

Maximum DSM's per phase: DSM_{max}

PSO Parameters:

- Swarm Size
- Generations



Top level decision vector:

...formulation continued

$$\mathbf{X} = [P_1, P_2, \ldots, P_{n-1}, D_1, D_2, \ldots, D_n]$$

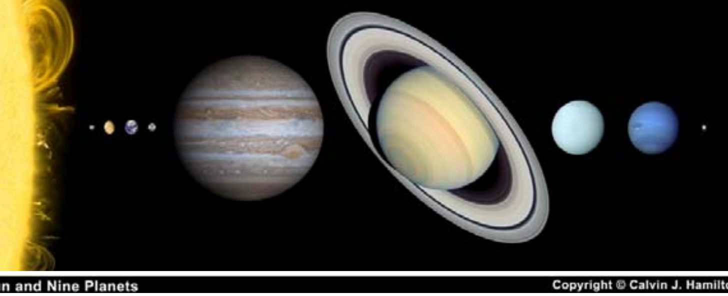
$$P_{1..n} \in [0, 9]$$

$$D_{1..n} \in [0, DSM_{max}]$$

Lower level: (objective function of the top level)

Pruning algorithm, followed by optimisation trying to minimize:

$$\sum \Delta V$$



University of Reading

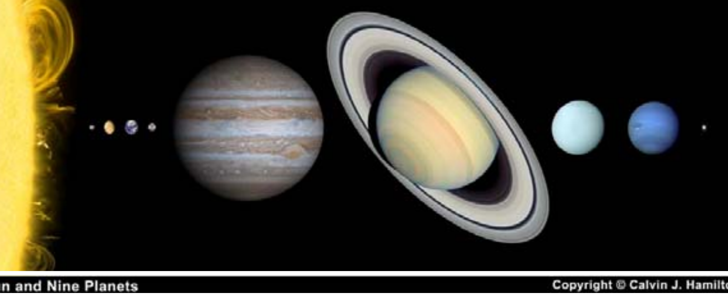
UNIVERSITY of GLASGOW

Case Study 1: Going from Earth to Mercury

Optimize the planetary sequence between Earth and Mercury
Find optimal DSM parameters for each phase
Minimize the total Delta-V

$P_{depart} = 3$
$P_{dest} = 1$
$n = 3$
$DSM_{max} = 1$

Restrict solution to planets between Mars and Mercury



Going from Earth to Mercury (cont.)

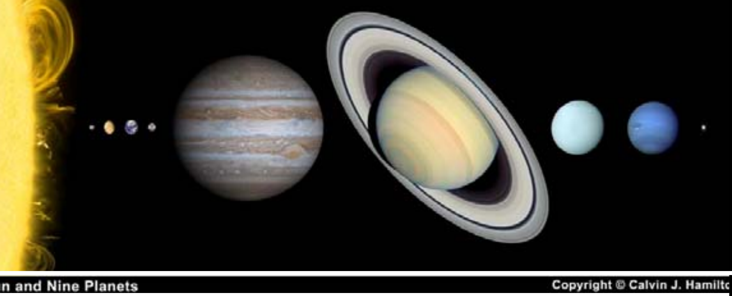
No insertion manoeuvre

Sequence	DSM	Delta-V
E-E-V-Me	110	1.175 km/s
E-E-V-Me	111	1.746 km/s
E-E-M-Me	110	2.490 km/s
E-V-Me	01	2.838 km/s

With insertion manoeuvre

r_p = 2480 km
e = 0.6679

Sequence	DSM	Delta-V
E-E-V-Me	000	6.9994 km/s
E-E-V-Me	001	8.1962 km/s
E-E-V-Me	111	8.6759 km/s
E-E-Me	11	9.7797 km/s



Case Study 2: GTOC 1 Impact Mission

Try to maximise the change in the semi-major axis of the asteroid

2001TW229

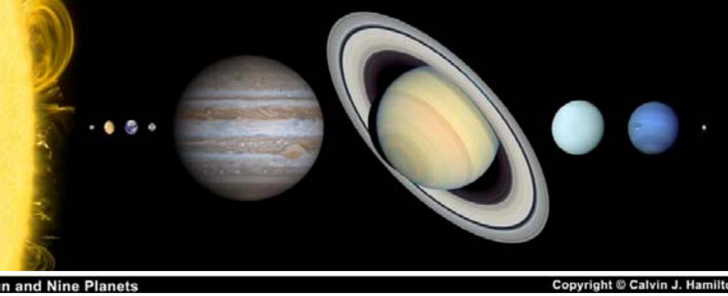
Top Level: Use PSO to optimise the integer variables characterising the sequence and presence of DSM’s

Lower Level (two sub-cases):

Initially prune to minimise the Delta V

Optimise the deflection of the asteroid

From published results we know that a retrograde transfer in the final phase maximises the deflection



Case Study 2: GTOC 1 Impact Mission

Best trajectory found by the sequence optimizer:

E-E-S-S-S-Ast

1 DSM in phases 1,2&3

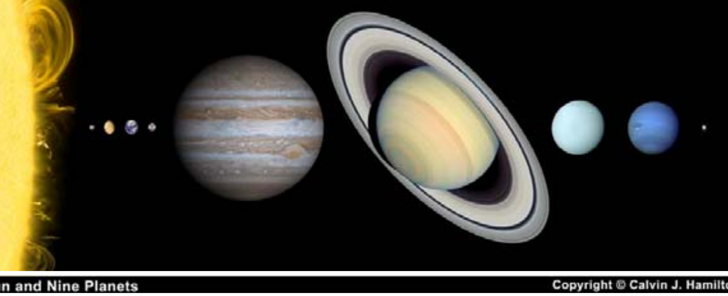
J = 997,644

Best solution presented in competition: (found by JPL)

E-V-E-E-E-J-S-J-Ast

J = 1,850,000

Solution contains low thrust arc

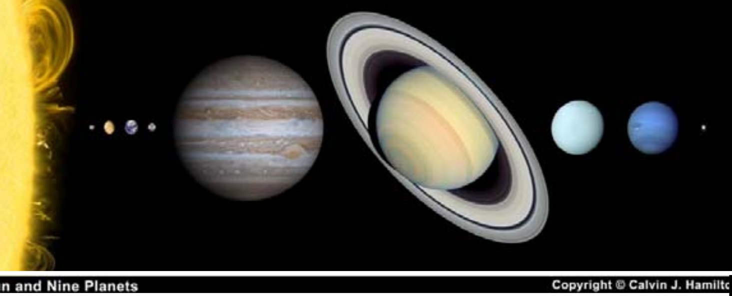


University of Reading

UNIVERSITY of GLASGOW

Some trajectories considered by the optimizer:

Sequence	DSM	J
E-E-S-S-S-Ast	11100	997644
E-E-E-V-V-Ast	11110	822401
E-E-J-S-J-Ast	11110	735174
E-Ast	1	410038
E-E-Ast	11	280457

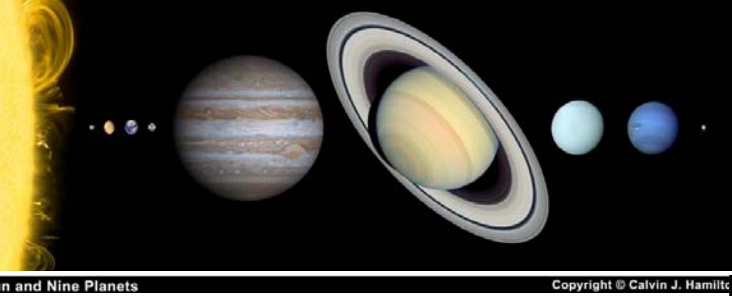


University of Reading

UNIVERSITY of GLASGOW

Concluding remarks from Reading work

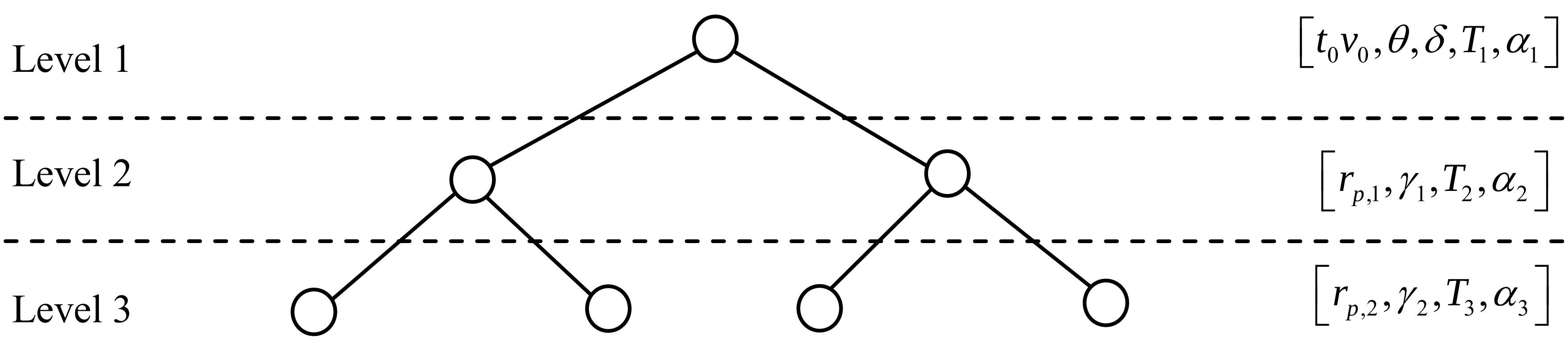
- GASP extended to DSM
- Pruning based on constrained local optimization.
- Clustering employed to identify feasible regions.
- Pruning algorithm scales linearly with the number of phases
- Two level optimization method proposed to optimise the phase sequence and number of DSM.



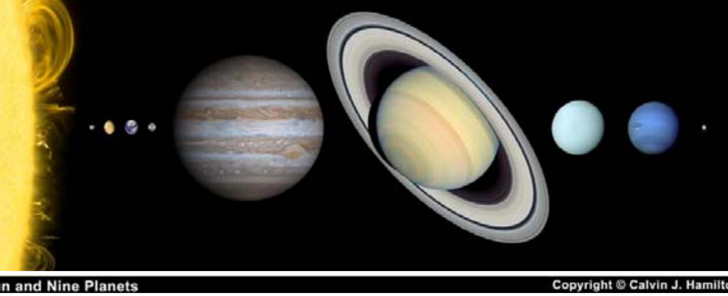
The Incremental Approach

MC10

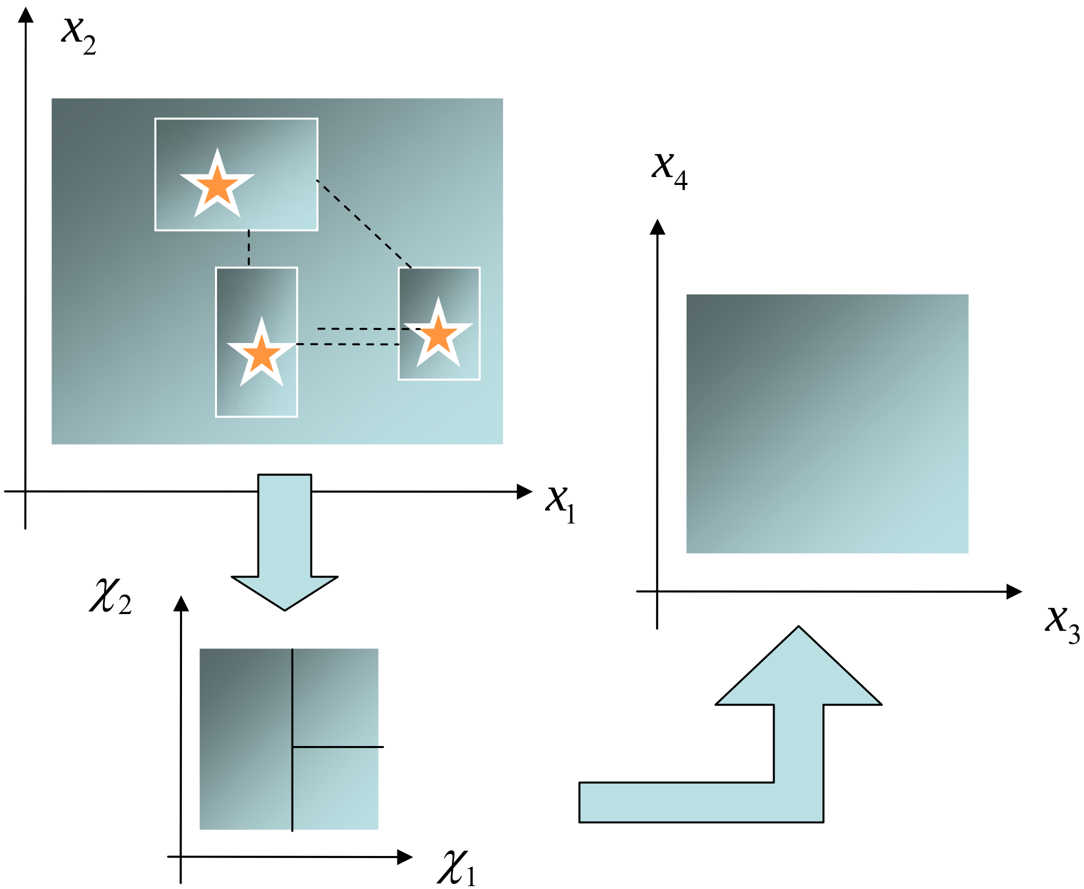
- Decomposition in subproblems (levels)
- Search for partial solutions at each level
- Incremental growth of the dimensions of the search space
- Incremental composition of the solution



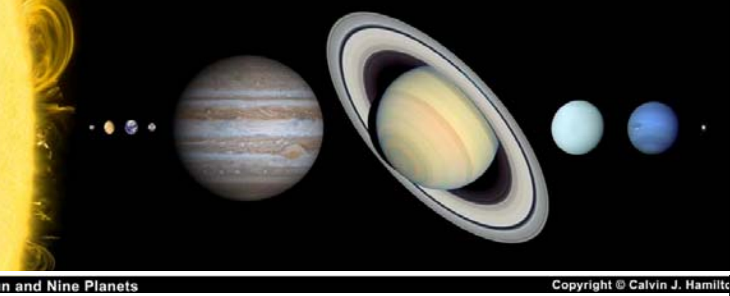
MC10 Changed figure colors
Matteo Ceriotti, 9/6/2007



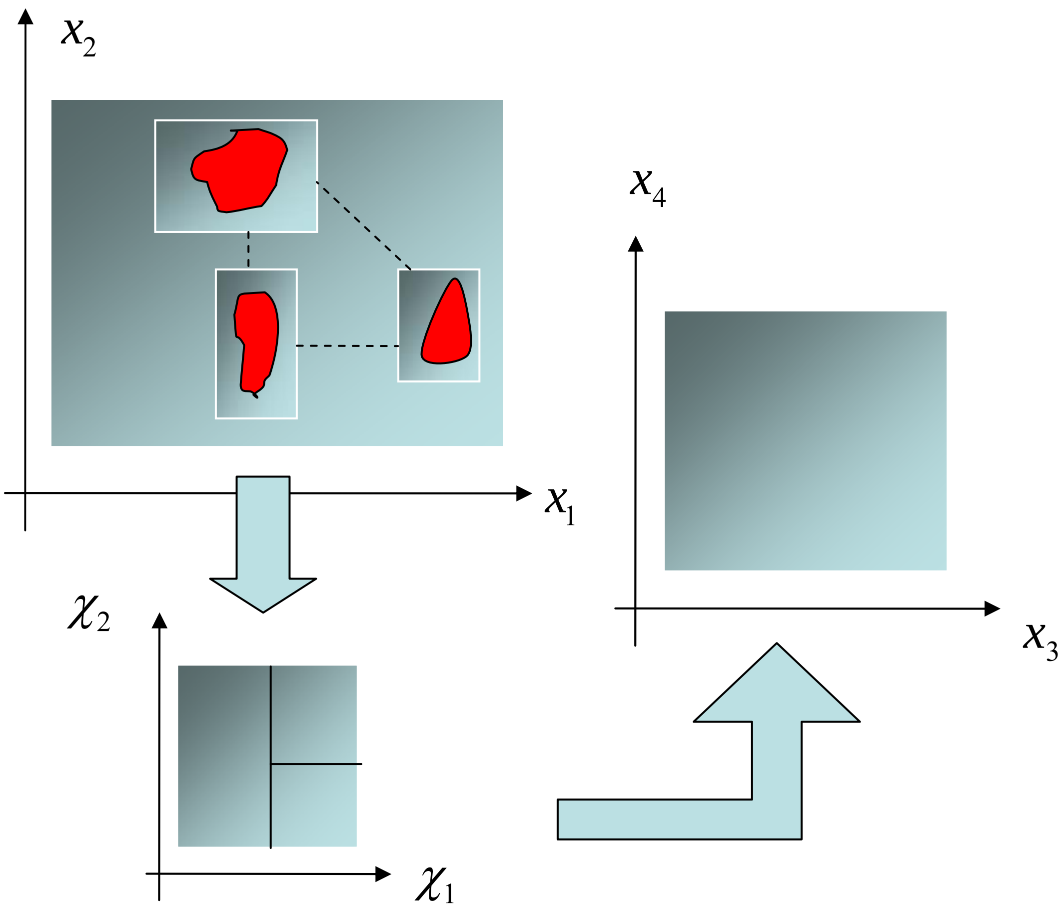
Incremental Approach IIIa



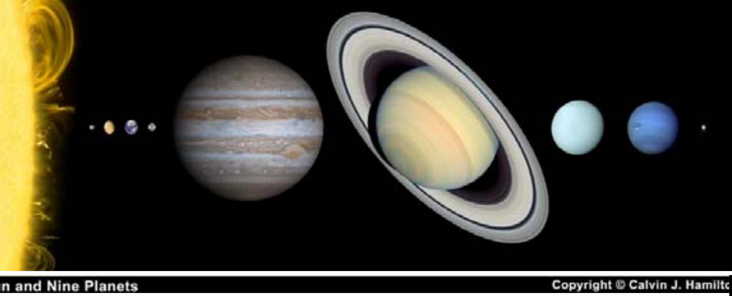
1. Decompose the problem into *M* subproblems
2. For each subproblem *k* (with *k=1,…,M*) define a pruning criterion f_k
3. Find a set of local minimisers for f_k in each subdomain D_k.
4. Build a box B_q around each minima
5. Build a new connected search space D_k^a at level *k* made of all the boxes B_q
6. Perform an optimisation on subdomain D_{k+1} keeping fixed the search space D_k^a



Incremental Approach IIIb



1. Decompose the problem into *M* subproblems
2. For each subproblem *k* (with k=1,…,M) define a pruning criterion f_k and a threshold f_t
3. Find all disconnected feasible sets X_q within subdomain D_k such that $f_k \leq f_t$ for all $\mathbf{x} \in X_q$.
4. Build a box B_q around each feasible set
5. Build a new connected search space D_k^a at level *k* made of all the boxes B_q
6. Perform an optimisation on subdomain D_{k+1} keeping fixed the search space D_k^a



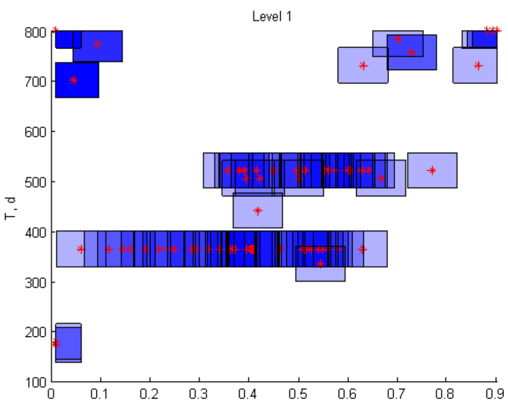
University of Reading

UNIVERSITY of GLASGOW

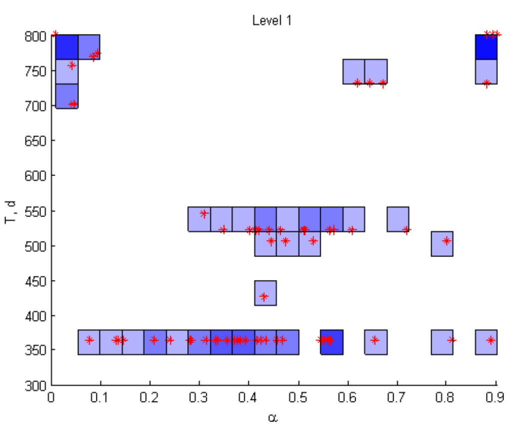
MC9

Box Identification

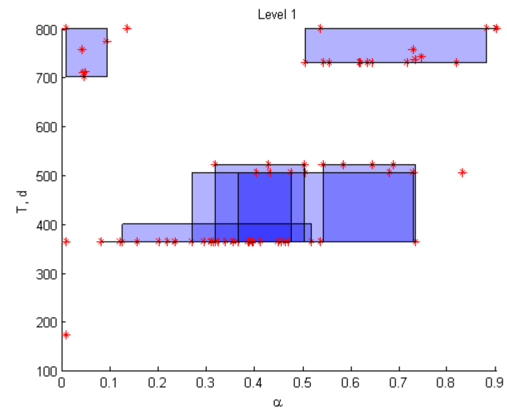
- 4 methods were studied



1

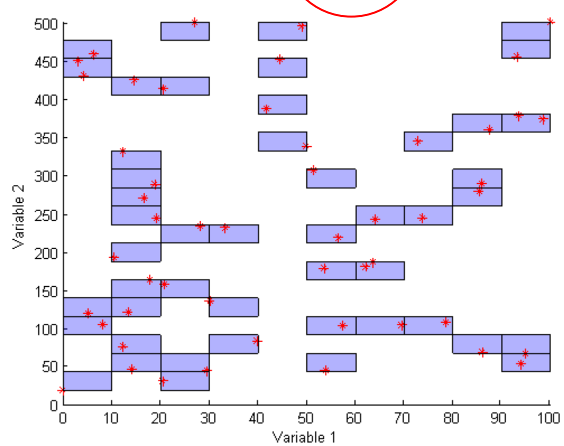


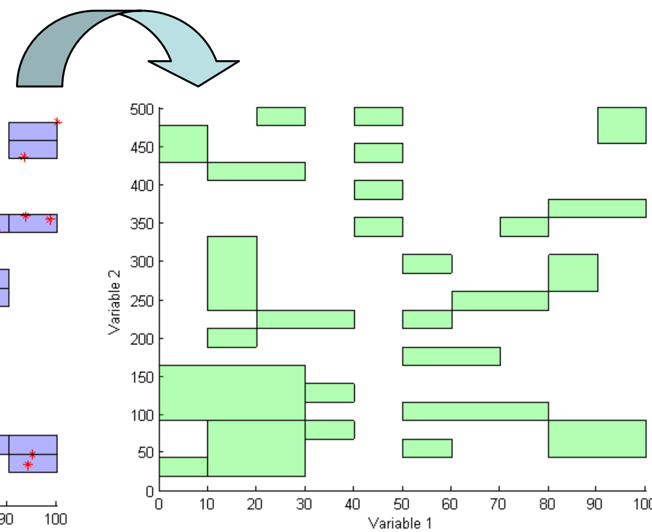
2



3

4

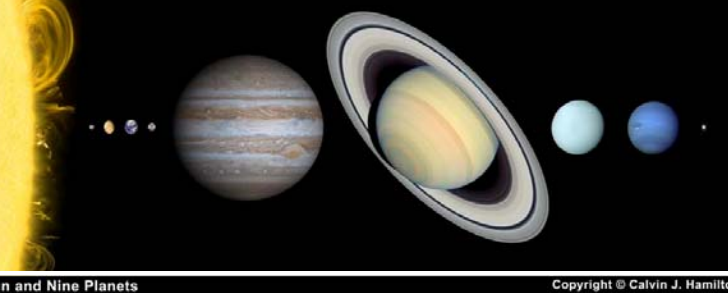




Slide 62

MC9 Just added

Matteo Ceriotti, 9/6/2007



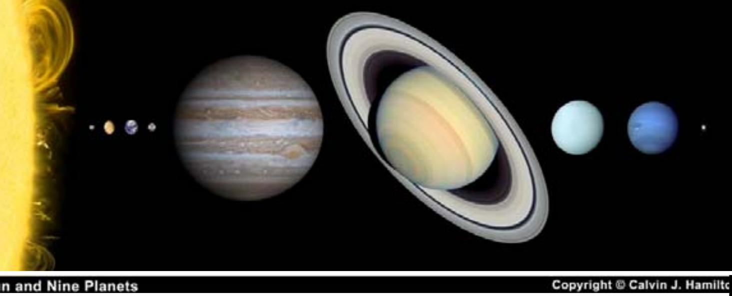
Affine Transformation and Box Collection

- Given a set of disconnected boxes B_q^k at level k
- We can define a biunivocal mapping function ϕ_k such that:

$$\phi_k : \bigcup_{q=1}^{Q(k)} B_q^k \rightarrow U^{m(k)} \qquad \phi_k^{-1} : U^{m(k)} \rightarrow \bigcup_{q=1}^{Q(k)} B_q^k$$

- Where $U^{m(k)}$ is the unit hypercube of dimension $m(k)$
- The inverse mapping requires $Q(k)$ operations for each level k, thus for M levels the number of operations is simply:

$$\sum_{k=1}^{M} Q(k)$$

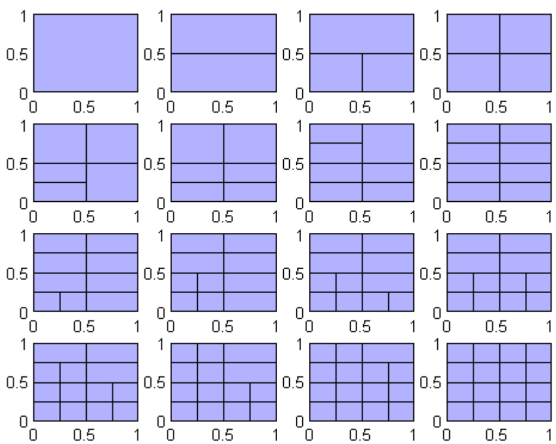


University of Reading

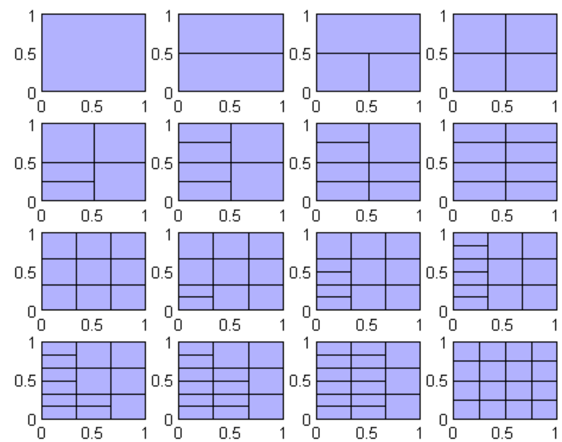
UNIVERSITY of GLASGOW

MC11

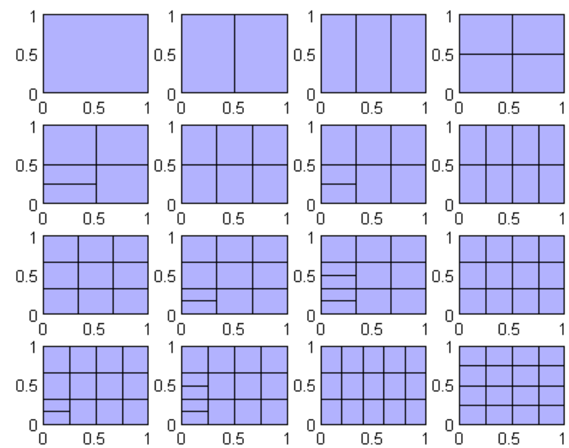
Affine transformation: different partitioning methods



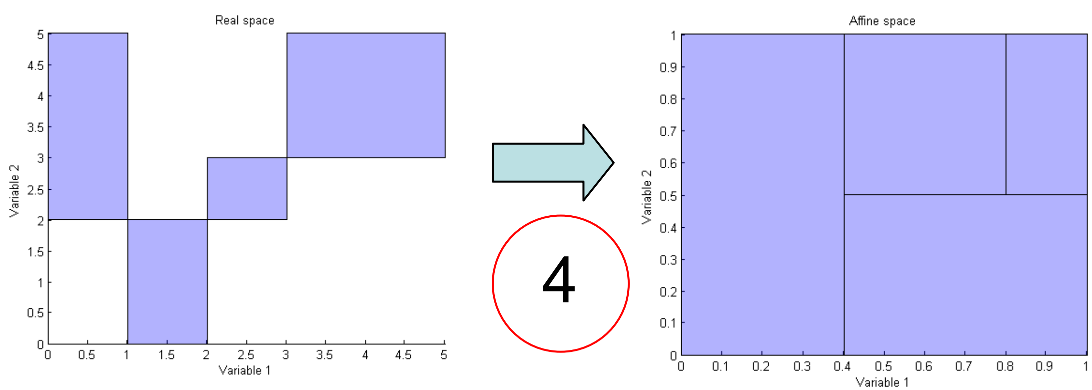
1



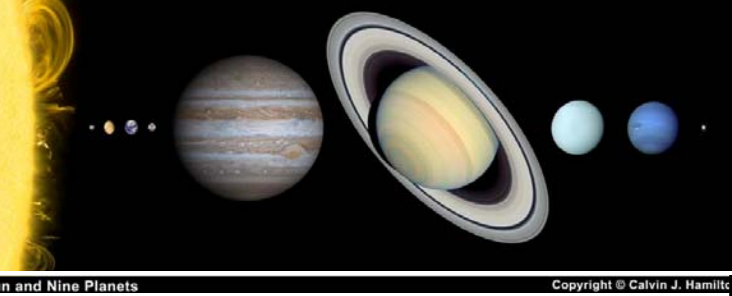
2



3



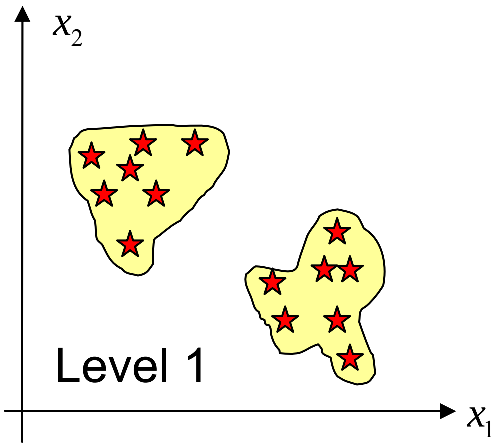
MC11 New slide
Matteo Ceriotti, 9/7/2007



MC12

Back pruning

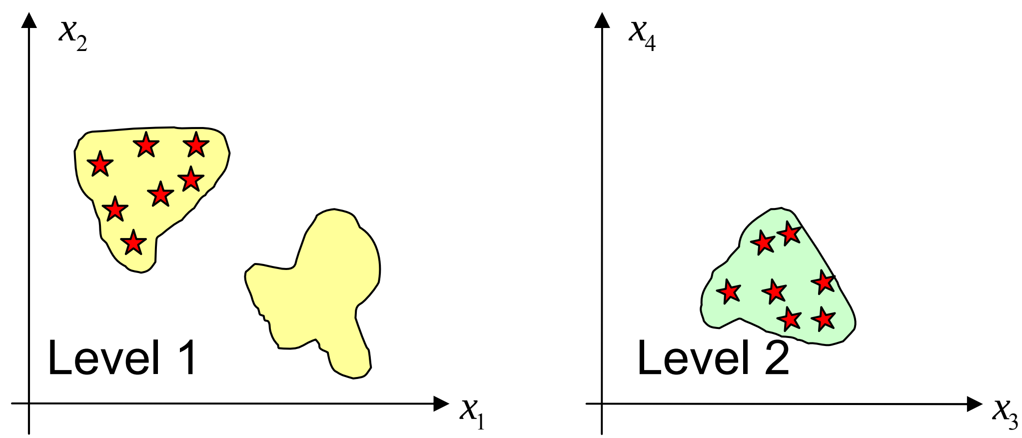
Pruning Level 1



Some of the regions, which were feasible at level *i*, then become unfeasible when adding one or more levels.

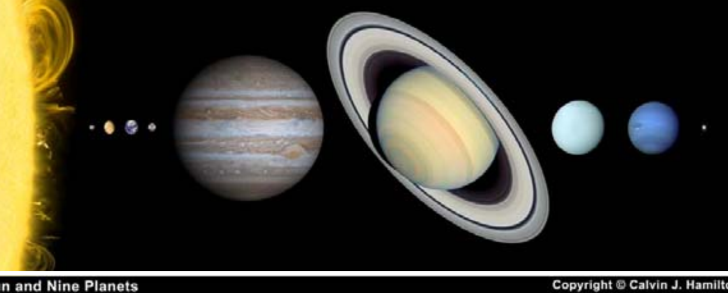
Back pruning further reduces the search space of the preceding levels

Pruning Level 2

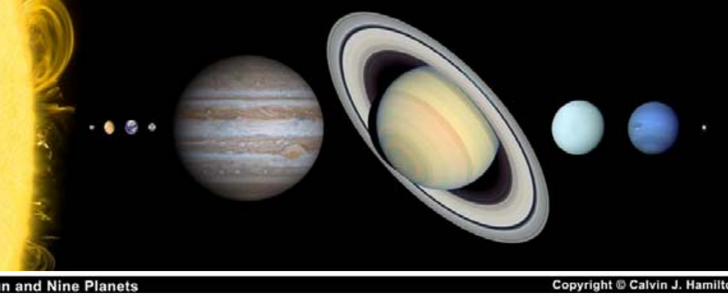


Slide 65

MC12 Added figures and text
Matteo Ceriotti, 9/7/2007



Testing the Pruning Approach



Preliminary Tests

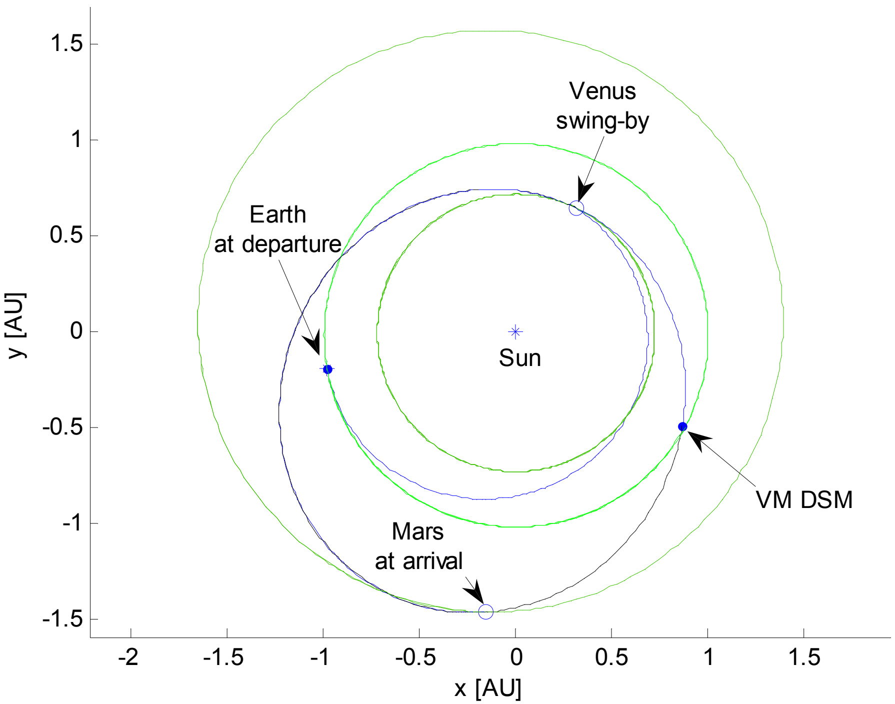
E-V-M

- No DSM on leg E-V
  - 2 variables on level 1
  - 4 variables on level 2
- $t_0 \in$[3650 MJD2000, +15 years]
- $TOF_1 \in$[50, 400] d
- $r_p \in$[1, 5] planet radii
- $T_2 \in$[50, 700] d

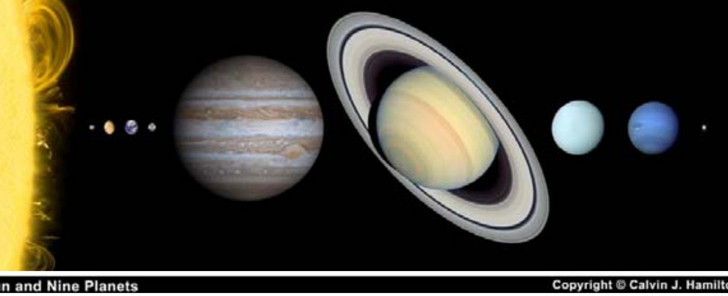
E-E-M

- DSM on each leg
  - 5 variables on level 1
  - 4 variables on level 2
- $t_0 \in$[3650 MJD2000, +15 years]
- v_0 =2 km/s
- $T_1 \in$[50, 700] d
- $r_p \in$[1, 5] planet radii
- $T_2 \in$[50, 700] d

E-V-M
All-at-Once Approach



- Objective: $v_0 + \Delta v_{DSM}$
- 500 random starting points + *fmincon*
- 494,233 total function evaluations
- Best solution: 2.98 km/s

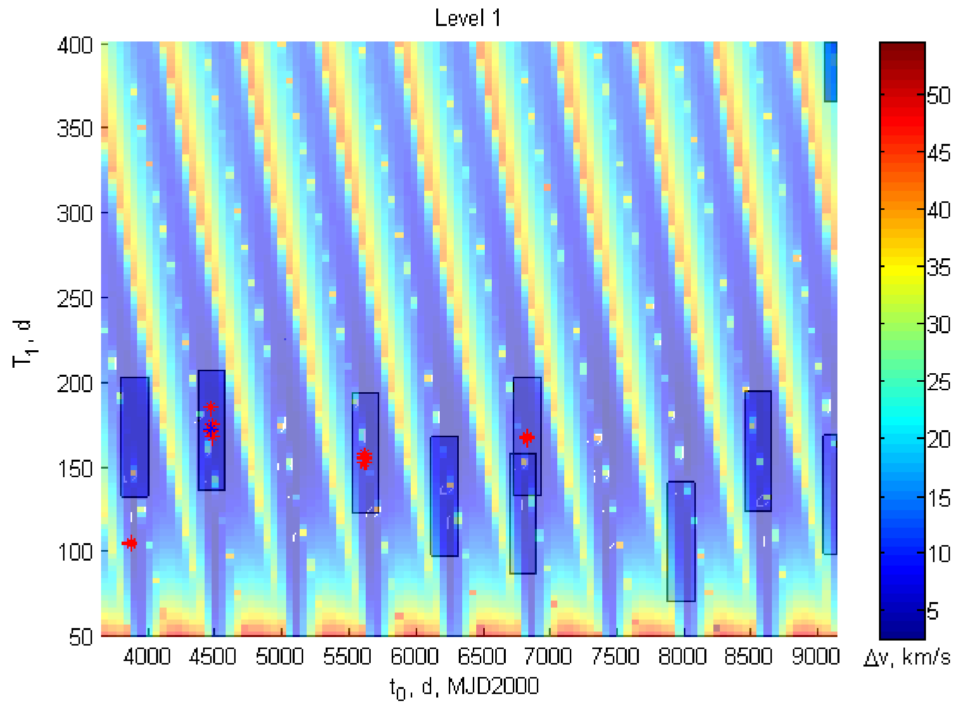


E-V-M
Comparison among Global Methods

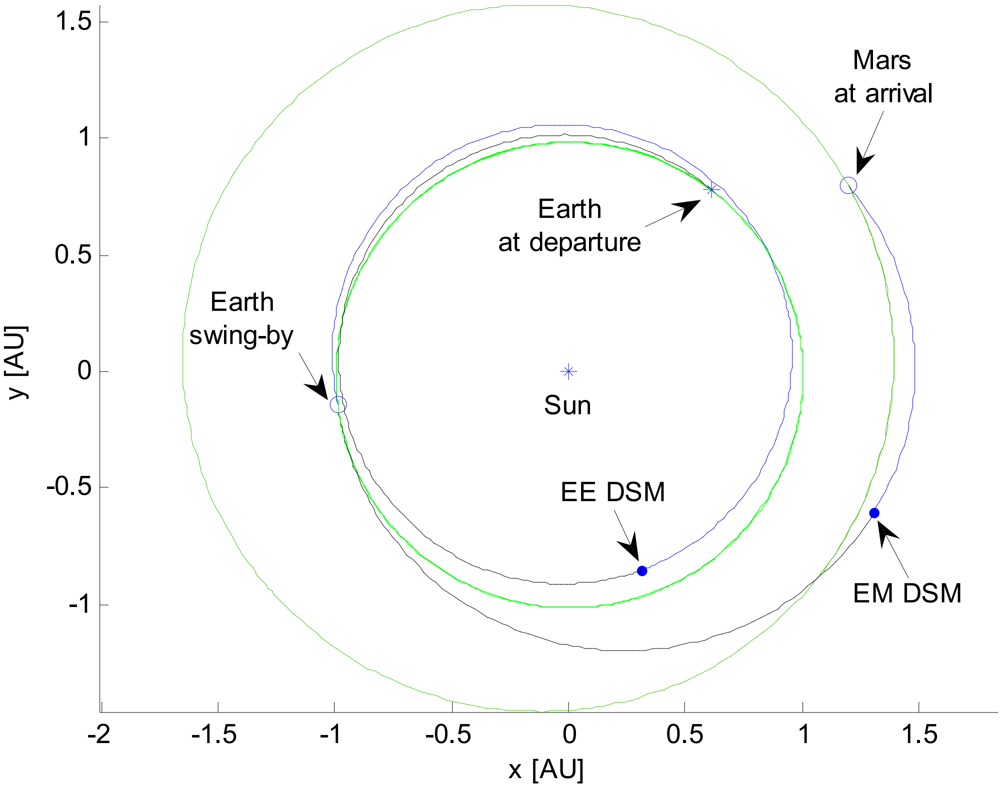
Solver	20000 evaluations	40000 evaluations	80000 evaluations
DIRECT [km/s]	4.3760	4.3730	4.3730
MCS [km/s]	6.7390	5.5240	5.4080
DEVEC, 200 runs			
< 3 km/s	6.5%	5.0%	7.0%
< DIRECT	99.5%	99.5%	99.5%
< MCS	100.0%	100.0%	100.0%
Multi-start, 200 runs			
< 3 km/s	2.5%	3.0%	3.0%
< DIRECT	97.0%	99.0%	98.5%
< MCS	100.0%	100.0%	100.0%
PSO, 200 runs			
< 3 km/s	2.0%	2.5%	7.5%
< DIRECT	71.5%	73.0%	78.5%
< MCS	100.0%	96.0%	93.0%

E-V-M
Incremental approach

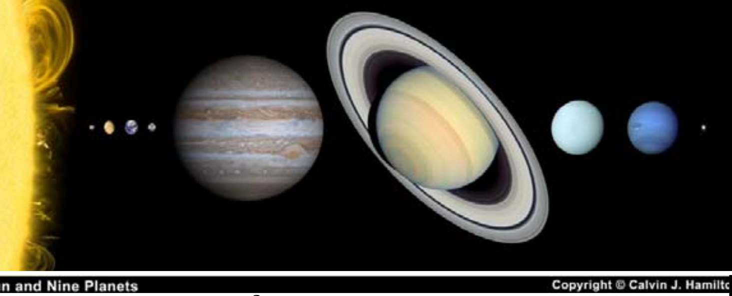
- 90% of the 20 best all-at-once solutions are included in one of the boxes
- The best all-at-once solution has been found
- 8827 function evaluations to find the best solution after pruning



E-E-M
All-at-Once Approach



- Objective: $\Delta v_{EE\text{-}DSM} + \Delta v_{EM\text{-}DSM}$
- 5000 random starting points + *fmincon*
- $9 \cdot 10^6$ total function evaluations
- Best solution: 0.326 km/s

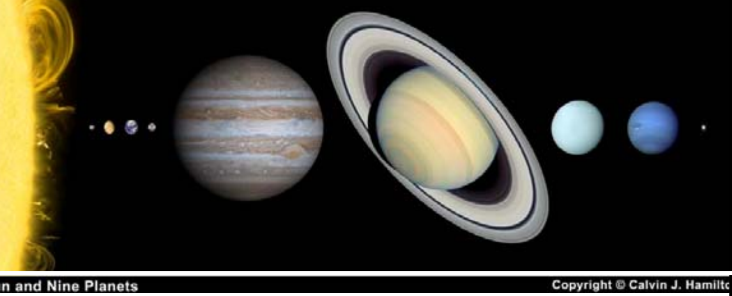


University of Reading

UNIVERSITY of GLASGOW

E-E-M
Comparison among Global Methods

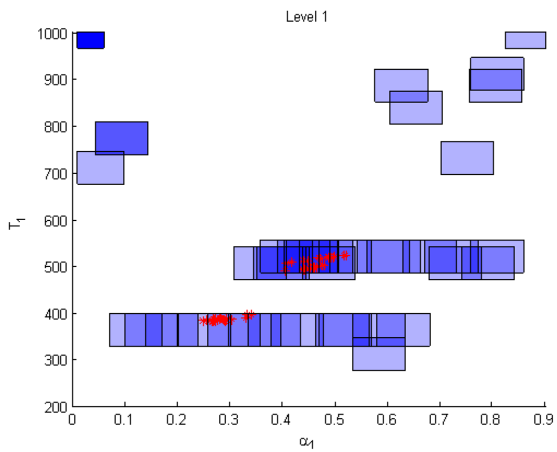
Solver	20000 evaluations	40000 evaluations	80000 evaluations
DIRECT [km/s]	2.7989	1.1870	1.1608
MCS [km/s]	1.2070	1.2070	0.9944
DEVEC, 300 runs			
< 0.33 km/s	0.0%	2.7%	8.0%
< DIRECT	69.7%	87.7%	85.7%
< MCS	100.0%	86.3%	85.7%
Multi-start, 300 runs			
< 0.33 km/s	0.3%	0.0%	0.7%
< DIRECT	100.0%	98.3%	98.7%
< MCS	94.7%	98.3%	96.0%
PSO, 300 runs			
< 0.33 km/s	0.7%	0.3%	0.0%
< DIRECT	100.0%	91.3%	76.3%
< MCS	84.0%	91.3%	71.3%

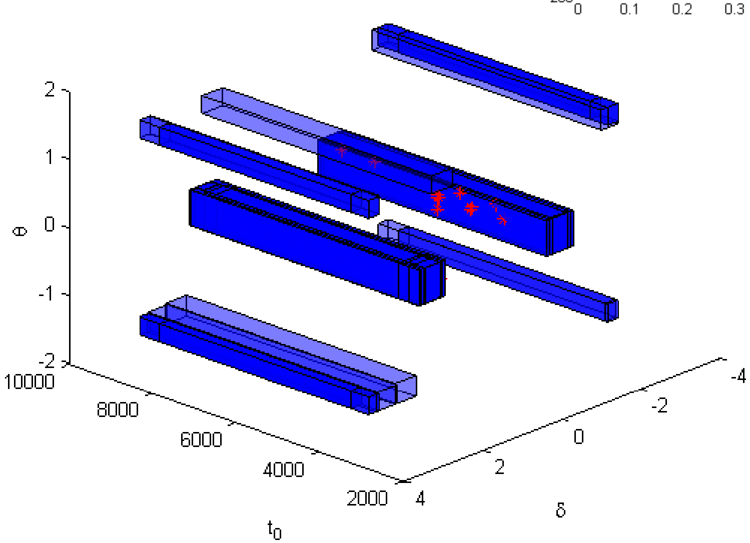


University of Reading

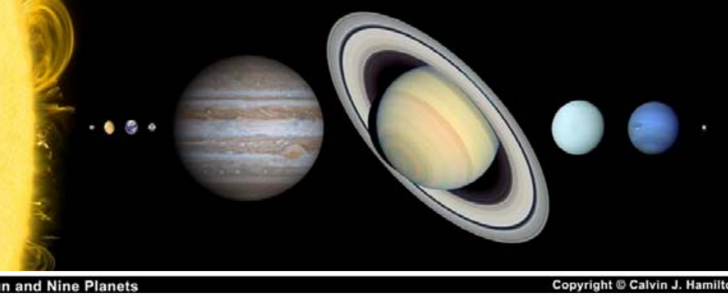
UNIVERSITY of GLASGOW

E-E-M
Incremental Approach



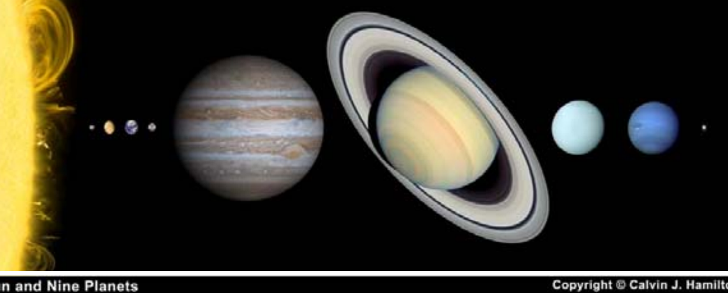


- 86% of the 50 best all-at-once solutions are included in one of the boxes
- The best all-at-once solution has been found
- 32544 function evaluations to find the best solution after pruning



Case Studies

- E-E-M with final orbit insertion
- The BepiColombo Mission
  - E-E-V-V-Me
  - E-V-V-Me-Me-Me
- Generation of Feasible Sequences for a NEO mission



E-E-M with final orbit insertion

- DSM on each leg
  - 5 variables on level 1
  - 4 variables on level 2
- $t_0 \in$[3650 MJD2000, +15 years]
- v_0 =2 km/s
- $T_1 \in$[50, 700] d
- $r_p \in$[1, 5] planet radii
- $T_2 \in$[50, 700] d

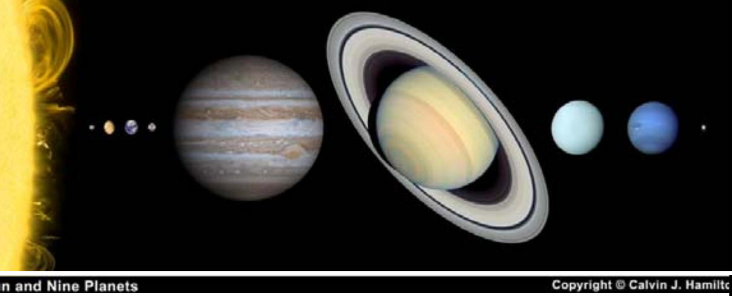
- Final orbit: r_p=3950, e=0.98

Incremental approach:

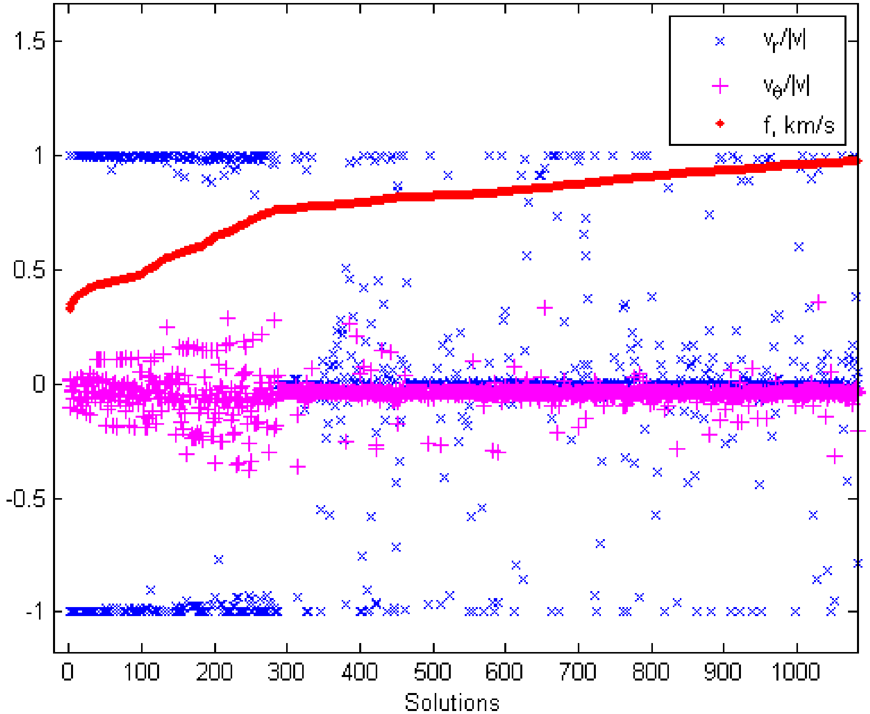
$$f_1 = \beta \frac{{v_\theta}^2 + {v_h}^2}{{v_r}^2} + \Delta v_1$$

$$f = \Delta v_1 + \Delta v_f$$

Starting points	
Level 1	Level 2
30	20

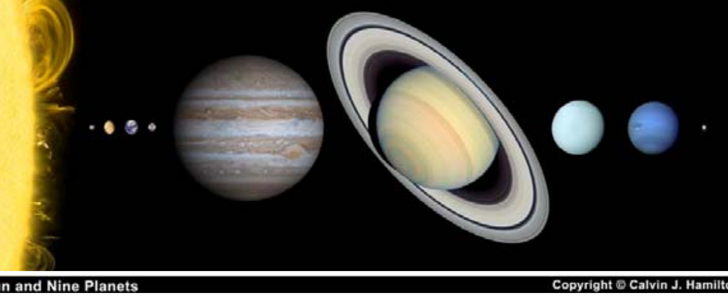


The RST Components



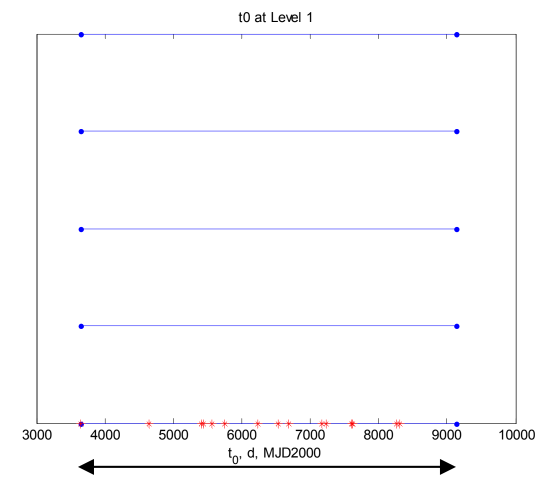
$$f_i = \beta \frac{v_\theta^{\;2} + v_h^{\;2}}{v_r^{\;2}} + \sum_{k=1}^{i} \Delta v_k$$

- Used for legs with resonant swing-bys
- Necessity to prune on the basis of the velocity before the swing-by



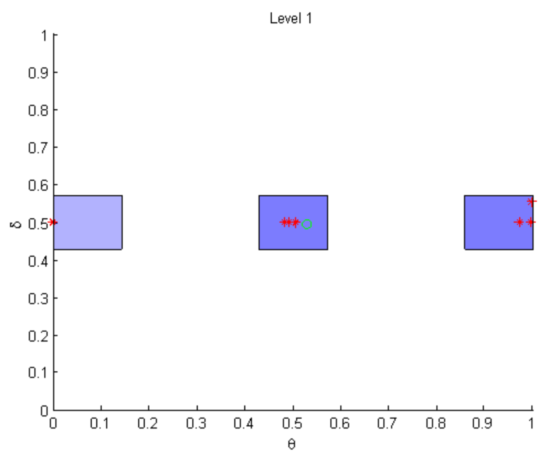
E-E-M with final orbit insertion
Pruning of level 1

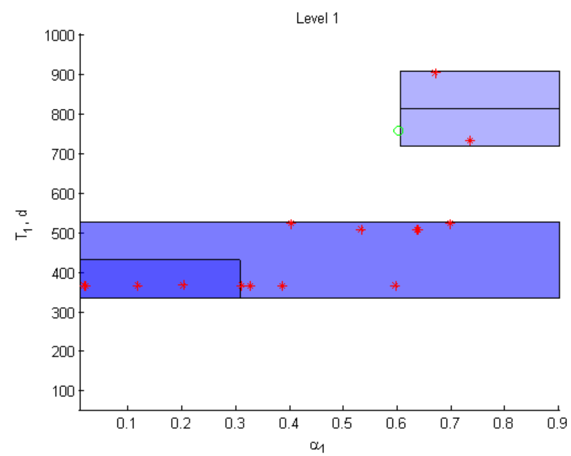
Level under pruning	Box edges on level 1
1	5478.75 d (1) 0.1429 (1/7) 0.1429 (1/7) 0.2967 (1/3) 95 d (1/10)

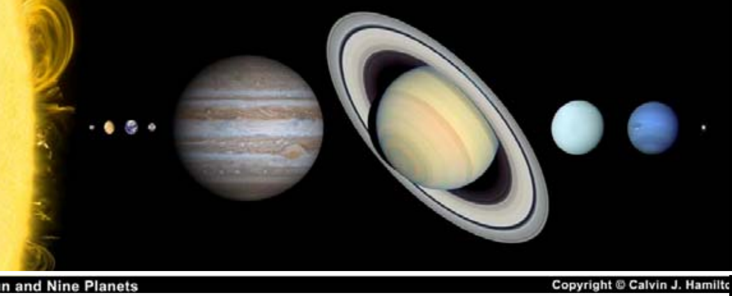


All launch window

Threshold = 0.5 km/s





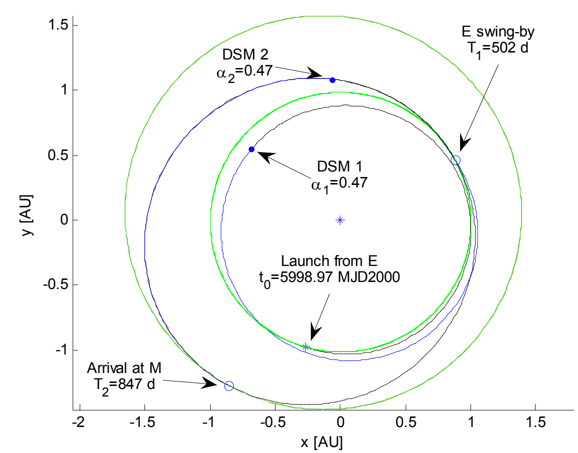


University of Reading

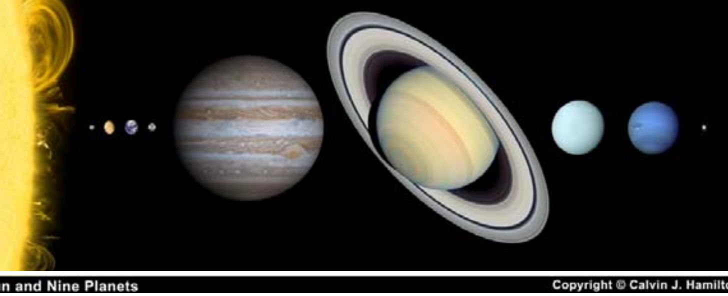
UNIVERSITY of GLASGOW

E-E-M with final orbit insertion Results

	Average function evaluations	Average best Δv [km/s]	Best Δv standard deviation [km/s]
DE all-at-once	200070	1.591	0.136
PSO all-at-once	200000	1.556	0.238
fmincon multi-start all-at-once	210217	1.268	0.137
Incremental	6097, 18519	1.171	0.081



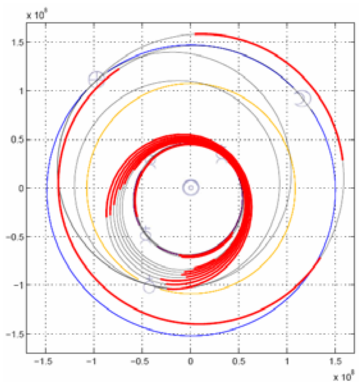
Δv=1.08 km/s



University of Reading

UNIVERSITY of GLASGOW

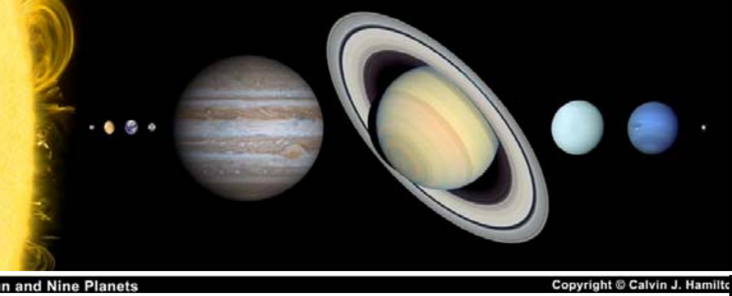
E-E-V-V-Me



Starting points			
Level 1	Level 2	Level 3	Level 4
100	100	100	200

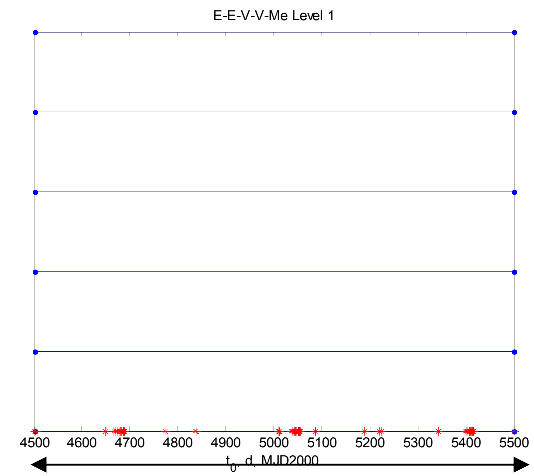
6.5 revs →

	Lower bound	Upper bound	Level
t_0 [d, MJD2000]	4500	5500	
θ	0	1	
δ	0	1	1
α_1	0.2	0.9	
T_1 [d]	350	600	
γ_1 [rad]	-π	π	
$r_{p,1}$ [planet radii]	1	5	
α_2	0.01	0.99	2
T_2 [d]	300	450	
γ_2 [rad]	-π	π	
$r_{p,2}$ [planet radii]	1	5	3
α_3	0.01	0.99	
T_3 [d]	150	300	
γ_3 [rad]	-π	π	
$r_{p,3}$ [planet radii]	1	5	4
α_4	0.595	0.733	
T_4 [d]	750	850	



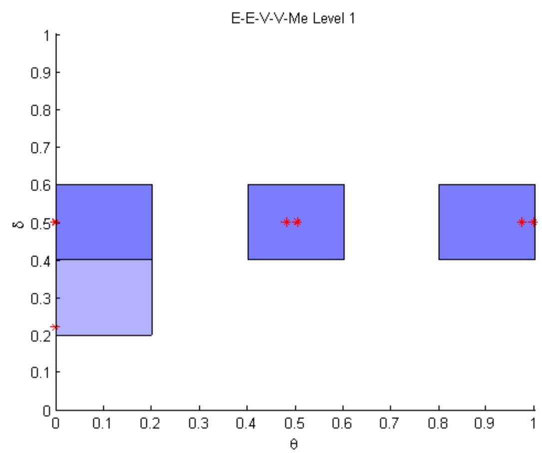
E-E-V-V-Me
Pruning of level 1

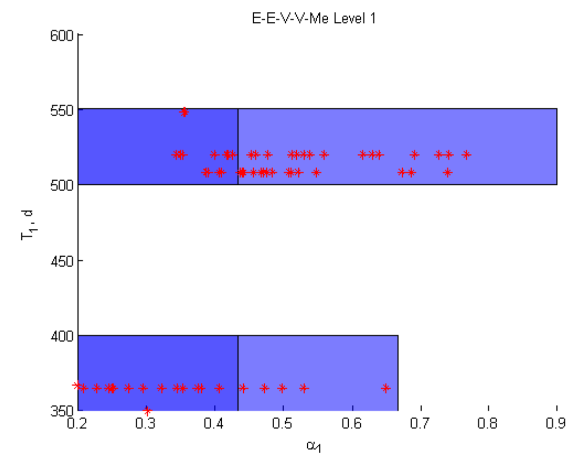
Level under pruning	Box edges on level 1
1	1000 d (1) 0.2 (1/5) 0.2 (1/5) 0.233 (1/3) 50 d (1/5)

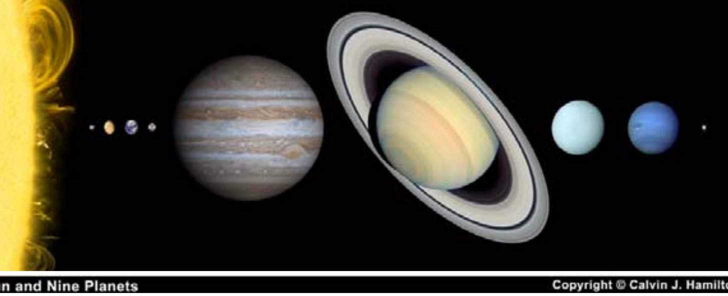


All launch window

Threshold = 1 km/s

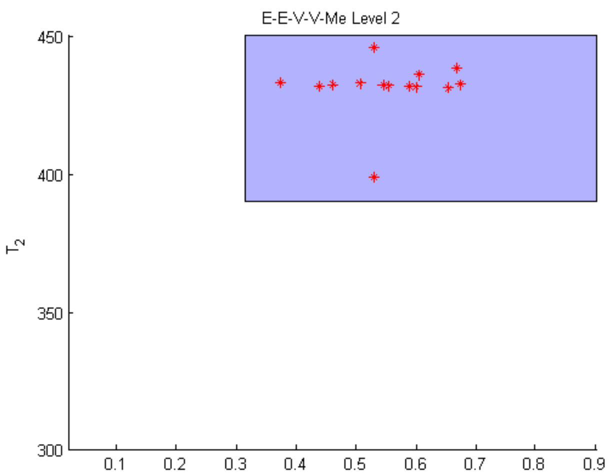


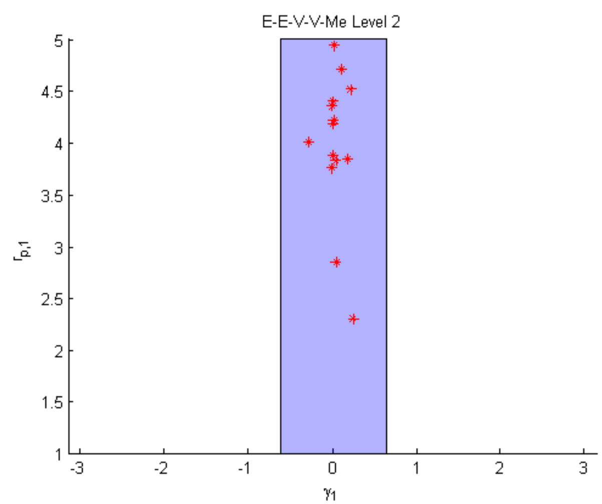


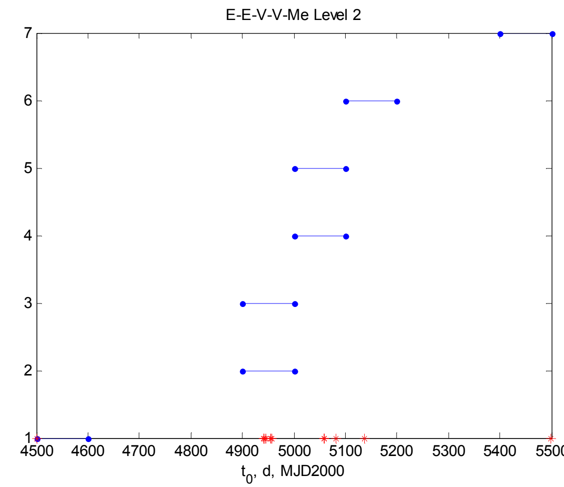


E-E-V-V-Me Pruning of level 2

Threshold = 1.1 km/s

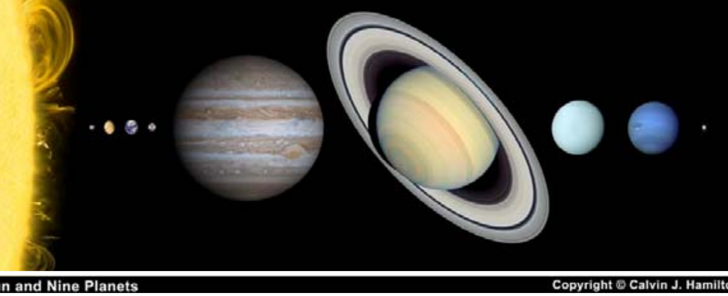




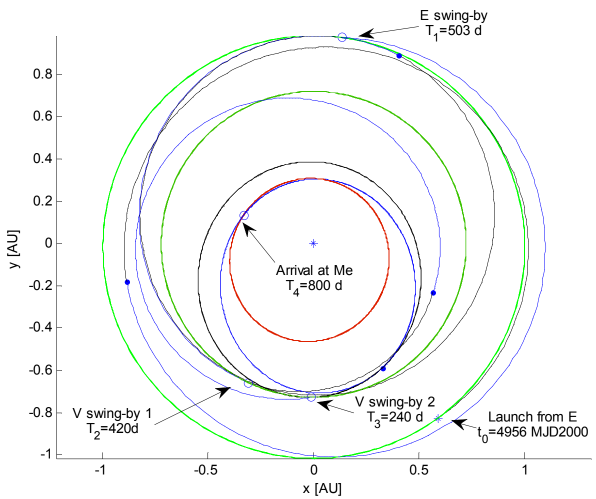


Back-pruning of level 1

Level under pruning	Box edges on level 1	Box edges on level 2
2	100 d (1/10) 0.2 (1/5) 0.2 (1/5) 0.233 (1/3) 50 d (1/5)	1.25 rad (1/5) 1.33 (1/3) 0.29 (1/3) 30 d (1/5)

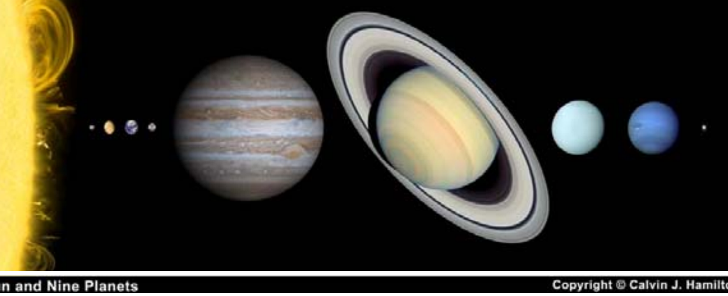


E-E-V-V-Me Results



Δv=4.55 km/s

	Average function evaluations	Total time for objective function evaluation [s]	Average best Δv [km/s]	Best Δv standard deviation [km/s]
DE all-at-once	400010	5842	8.456	0.444
PSO all-at-once	460000	6900	6.094	0.920
fmincon multi-start all-at-once	427499	6412	4.599	0.865
Incremental	24397, 96674, 184340, 154754	3625	3.89	0.739

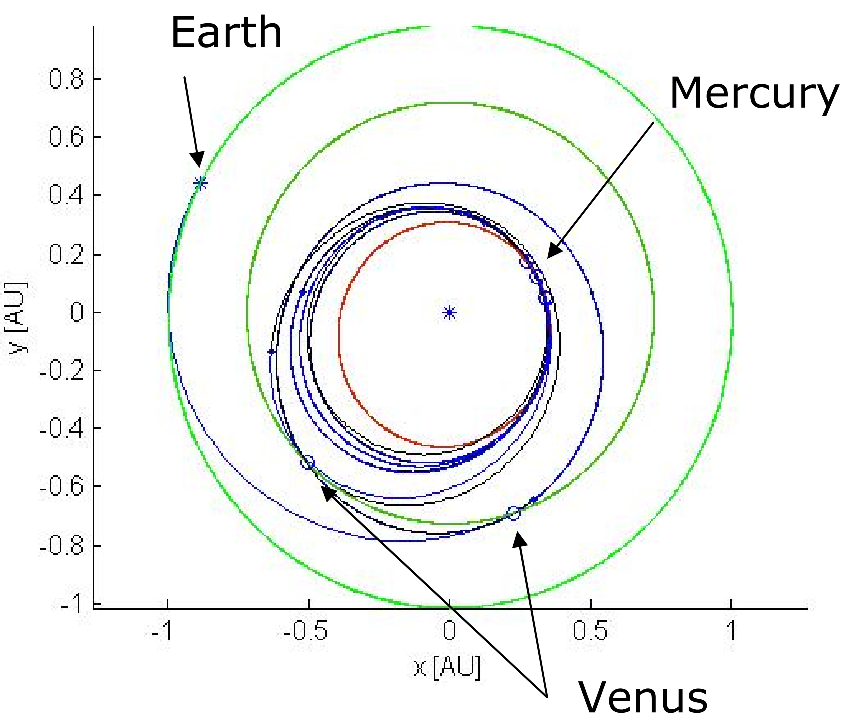


Feasible Set Approach: EVVMeMeMe Sequence

- 2000 days of search window [3457,5457]MJD2000
- No pruning on γ and r_p
- Pre-assigned resonance strategy, i.e. fixed number of revolutions per leg
- Boundaries on α and TOF function of the number of revolutions
- Special partial pruning criteria for Venus and Mercury incoming conditions

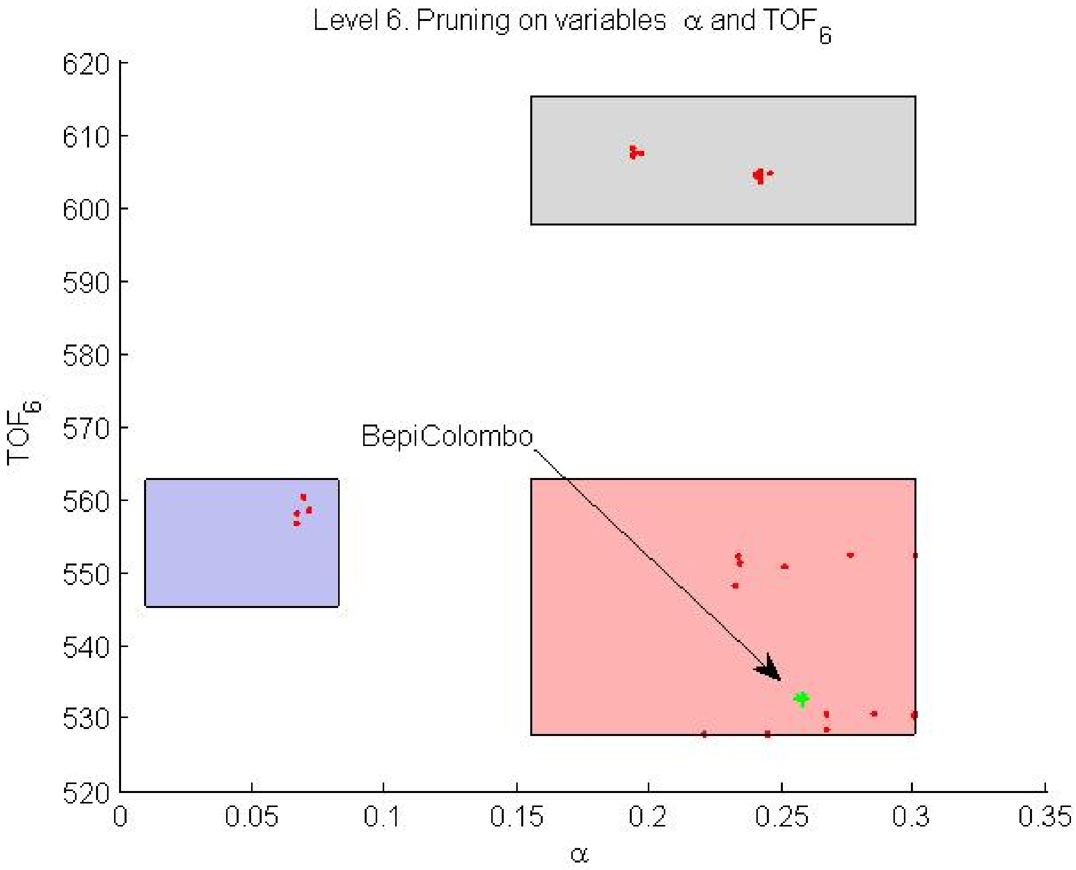
$$f_V = \frac{{v_\theta}^2 + {v_h}^2}{{v_r}^2} + \Delta v_0 \qquad f_i = \beta \frac{{v_\theta}^2 + {v_h}^2}{{v_r}^2} + \sum_{k=1}^{i} \Delta v_k$$

Feasible Set Approach: EVVMeMeMe Sequence



- The best sampled solution during the pruning has a total Δv=6.5km/s with a v_{inf}=4km/s at Mercury and v_{inf}=3.869km/s at launch.
- **Not a local minimum just a sample in the pruned space**
- BepiColombo has a total Δv=4.08km/s with a v_{inf}=3.44km/s at Mercury and v_{inf}=3.762km/s at launch
- 225000 function evaluations (about 45 minutes on a centrino 2GHz).
- Good repeatability of the pruning

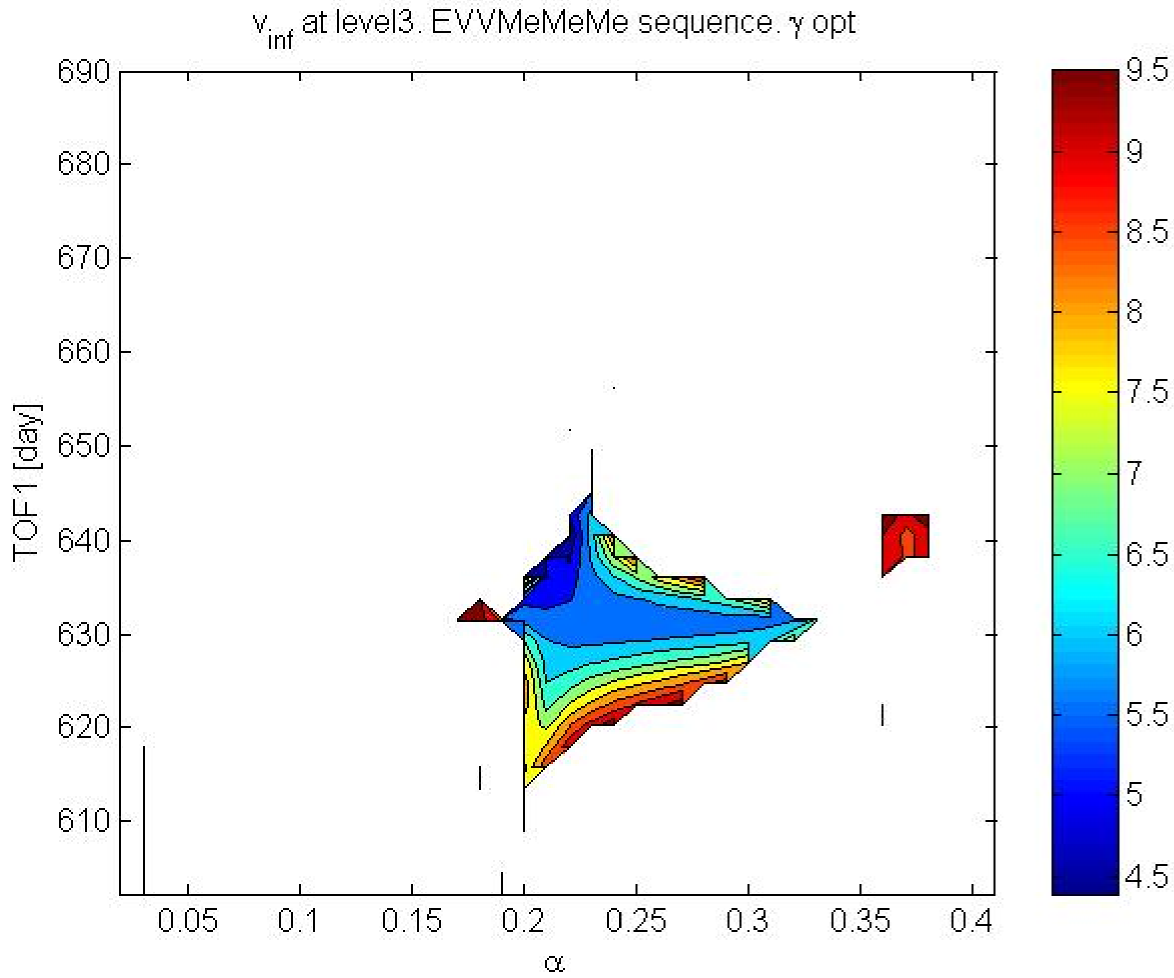
E-V-V-Me-Me-Me Pruning: Feasible Set Approach

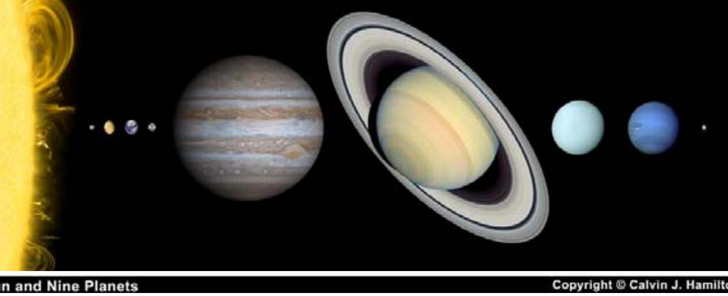


University of Reading

UNIVERSITY of GLASGOW

E-V-V-Me-Me-Me Pruning: Search Space Analysis





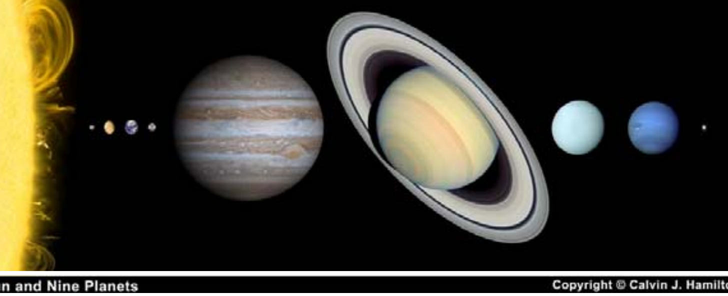
Characterisation of the Search Space

- At each level *k* we define a partial objective function:

$$f_k(\mathbf{x}) : D_k \subset \mathbb{R}^n \to \mathbb{R}$$

- The partial objective function might not be directly related to the objective function *f* of the whole problem and is uses solely to prune the search space D_k

- Definition of characterisation: at each level *k* we want to identify all the minima for f_k or all the regions that are feasible according to the condition $f_k \leq f_t$



- a local minimisers in D_k is:

$$\mathbf{x}_l \in D : \nabla f(\mathbf{x}_l) = 0 \wedge \mathbf{x}_l^T \mathbf{H}(\mathbf{x}_l)\mathbf{x}_l > 0$$

- and a ball containing a local minimiser *l* is:

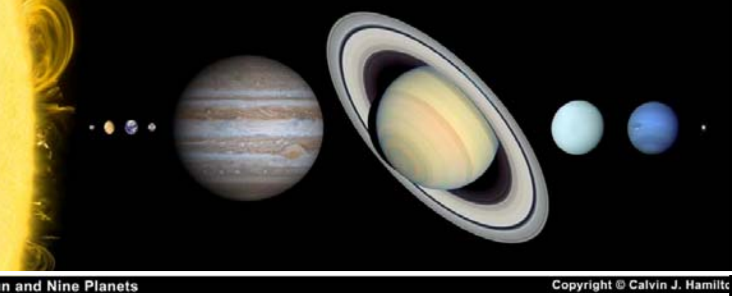
$$X_{\varepsilon,l} = \left\{ \mathbf{x} \mid \mathbf{x} \in D \wedge \left\| \mathbf{x} - \mathbf{x}_l \right\| < \varepsilon \right\}$$

- consider a set of grid points in D_k

$$X_N = \left\{ x_i \mid x_i \in D \wedge i = 1,\ldots,N \right\}$$

- then we have a sufficient grid to characterise D_k if for a given ε:

$$X_k \cap X_{\varepsilon,l} \neq \emptyset \quad \forall l$$



University of Reading

UNIVERSITY of GLASGOW

Algorithmic Complexity Analysis II

- The number of generated and stored boxes at each level k is $Q(k) \leq N_l$
- The total number of stored boxes for M levels is:

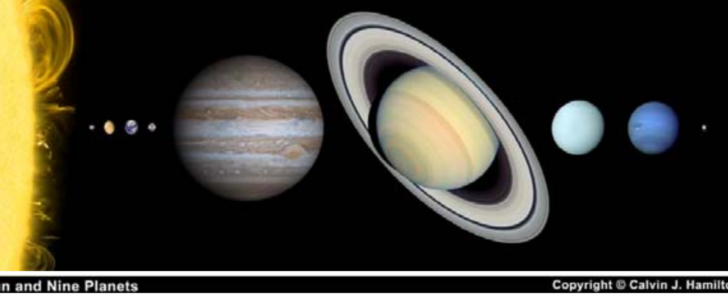
$$\sum_{k=1}^{M} Q(k)$$

- The affine transformation requires for each function evaluation $Q(k)$ operations to check the inclusion of the evaluated point
- The total number of operations for M levels is:

$$\sum_{k=1}^{M} Q(k)$$

- If $N_s(k)$ grid points (function evaluations) are required to characterise the search space at level k then the total number of grid points (function evaluations) is:

$$\sum_{k=1}^{M} N_s(k)$$

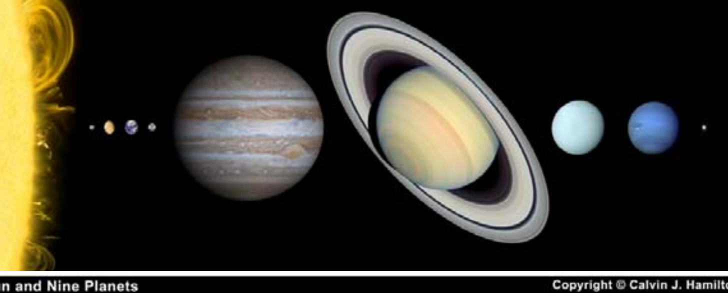


Trajectory Planning

Trajectory Planning

- The design of a sequence of GA,DSM and LT arcs is addressed by considering each trajectory as a scheduled sequence of actions.
- Each action has preconditions and post conditions

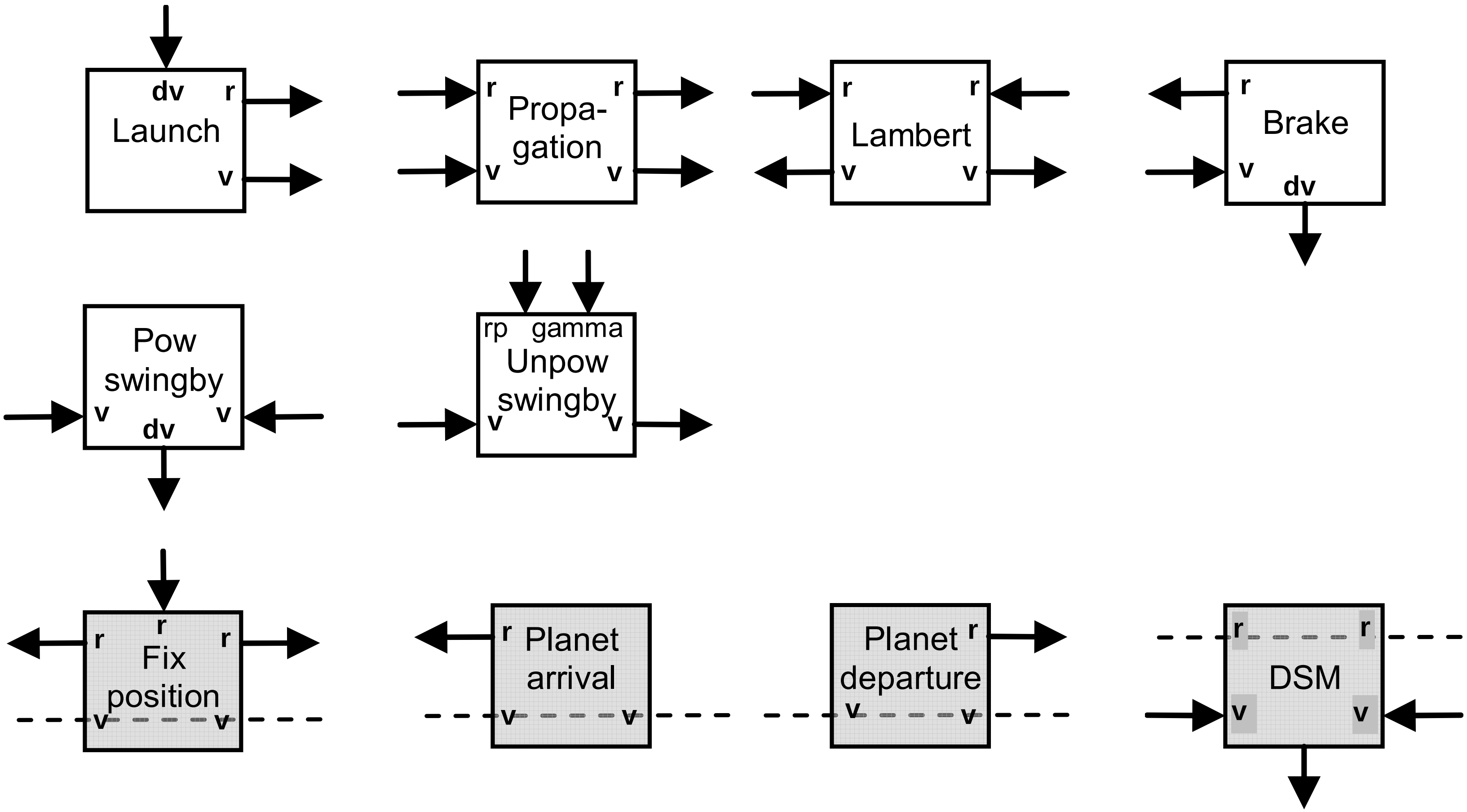
- First Pruning Heuristic
  - A complete trajectory is feasible if and only if all the pre and post conditions for all the actions composing the trajectory are true.

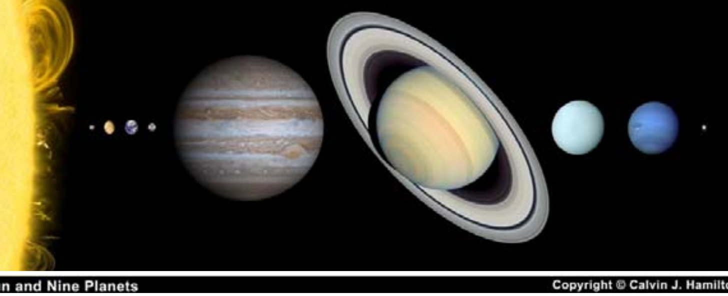


University of Reading

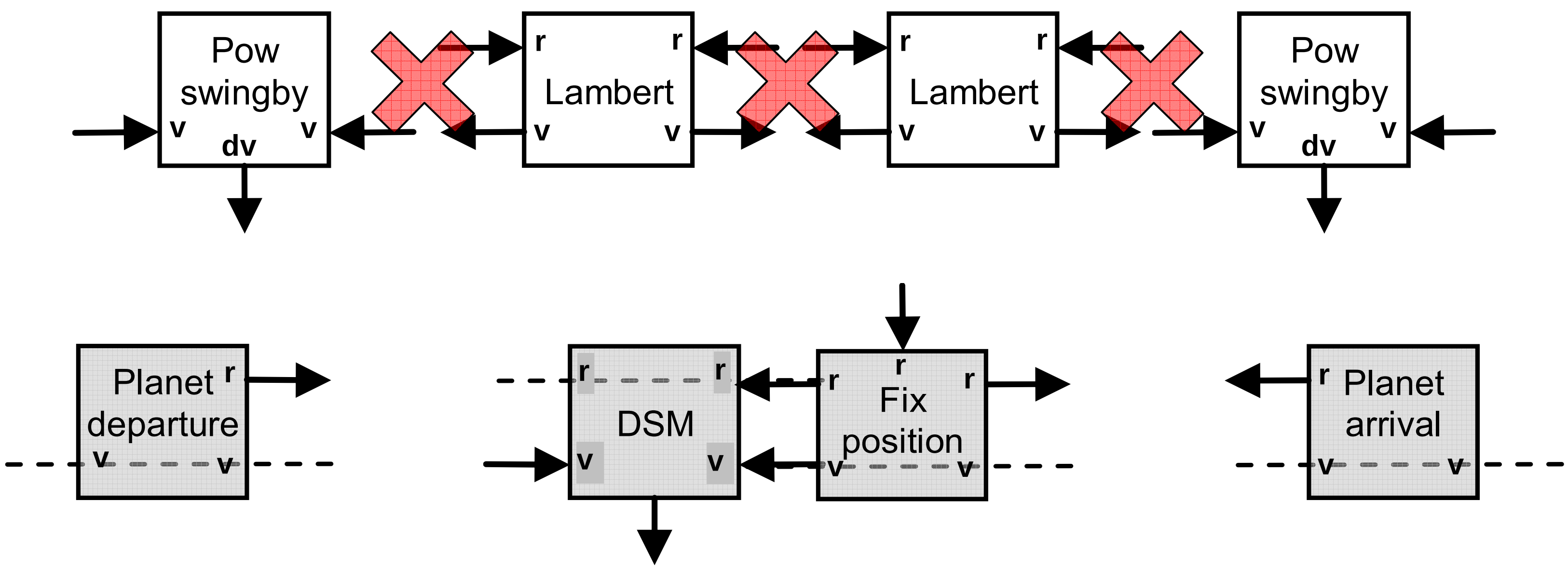
UNIVERSITY of GLASGOW

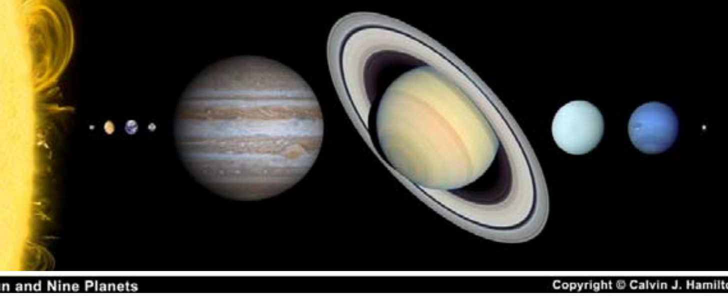
Block Structure Model: Blocks



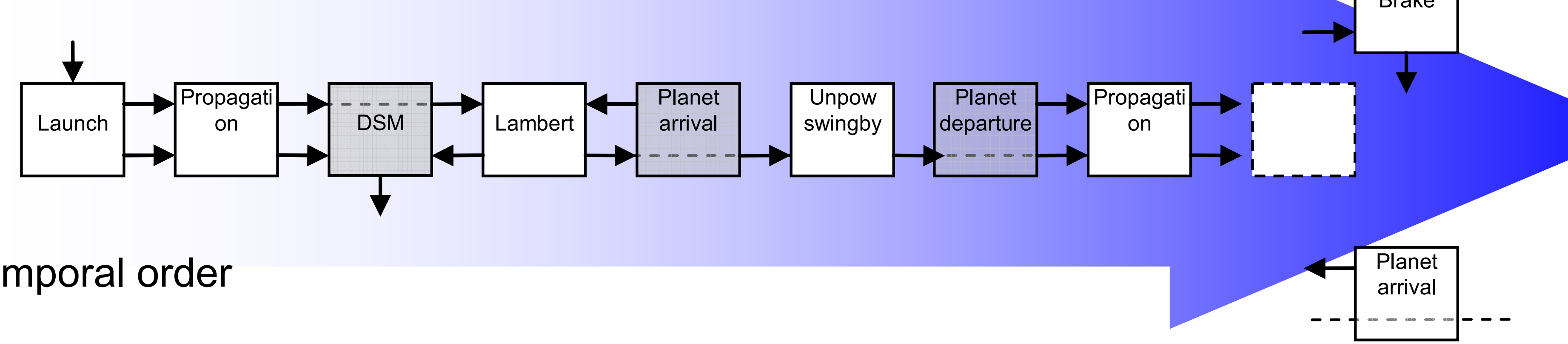


Feasibility

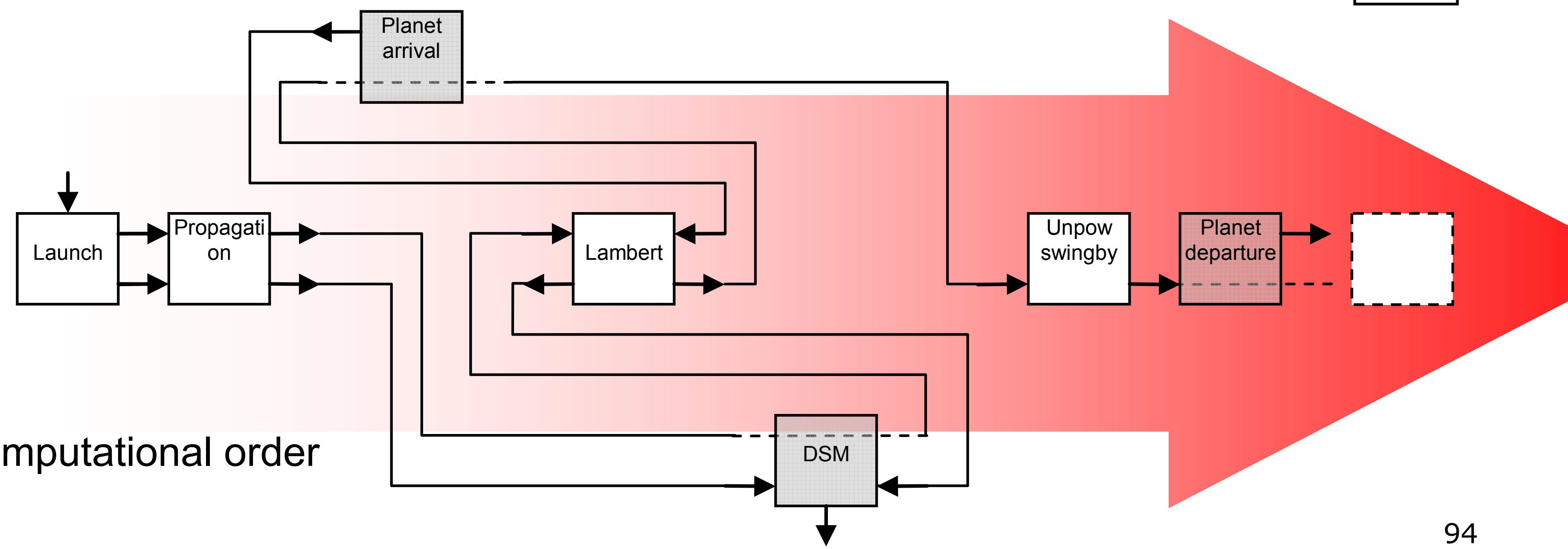




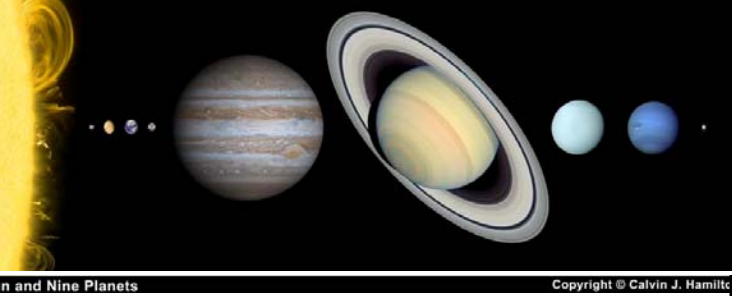
Model 1 with blocks



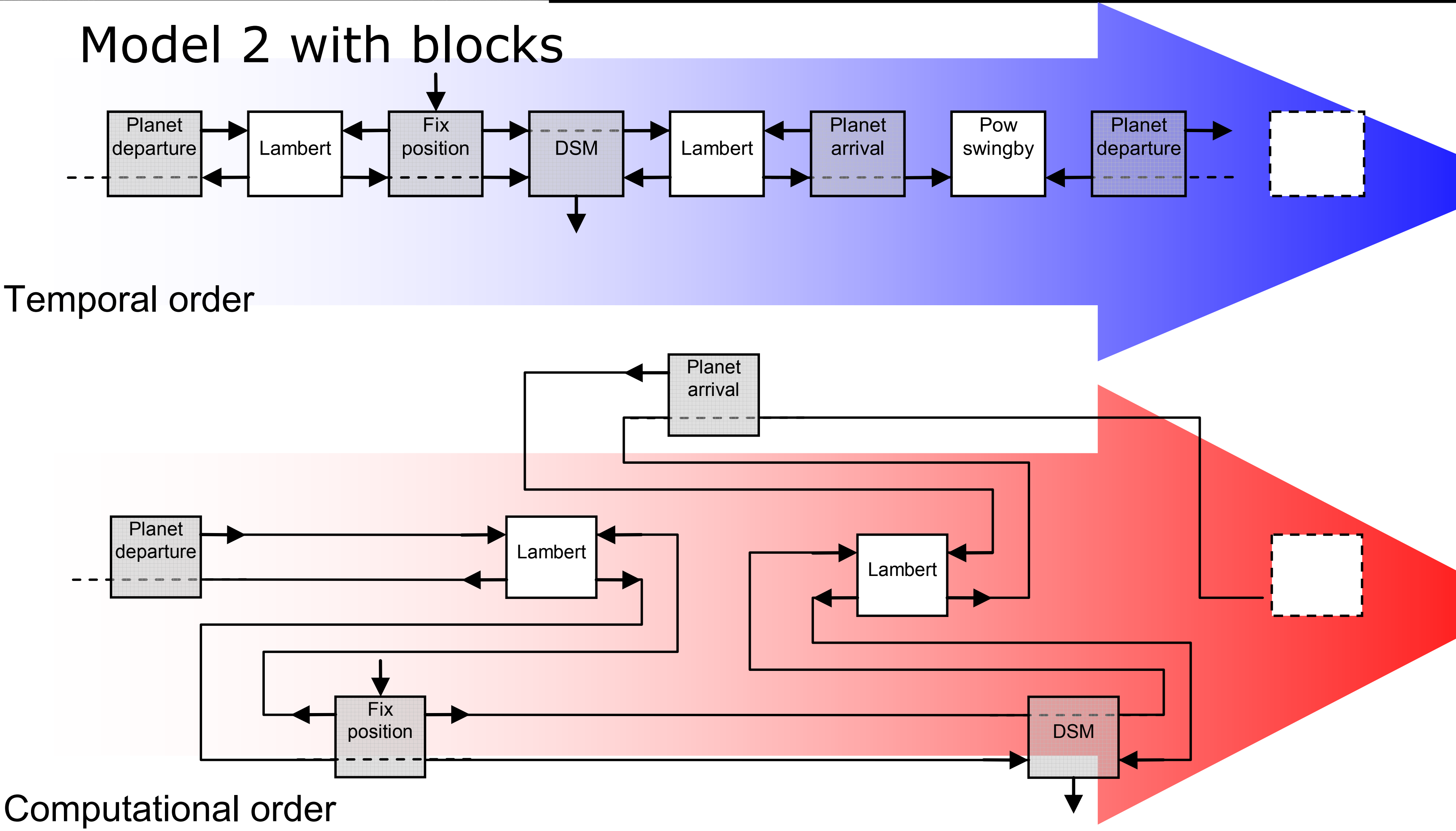
Temporal order

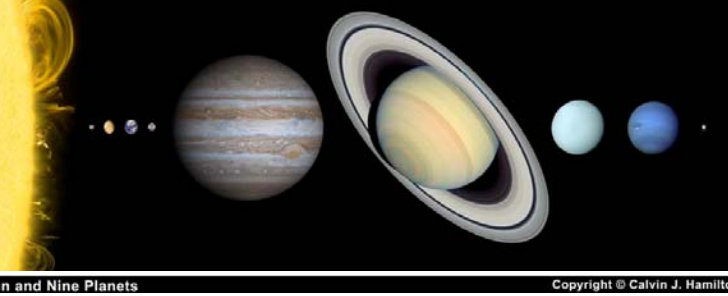


Computational order



Model 2 with blocks



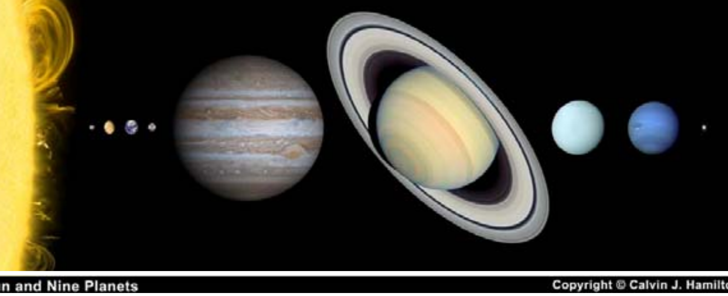


Generation of the Feasible Set

1. Define a set of pruning criteria P
2. Start at level i=0 with a departure action s_{01} for which ¬P
3. Add a level i=i+1
4. Add to each partial trajectory j an action s_{ij} the preconditions of which match the post conditions of s_{i-1j}.
5. If at level i multiple actions can be added create a partial trajectory for each action
6. Prune all partial trajectories that meet the pruning criteria
7. go to 3.

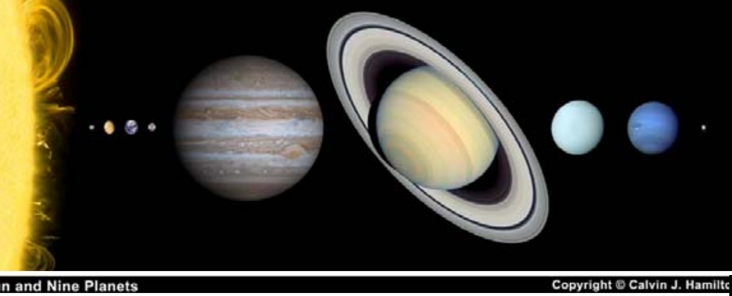
Automatic Sequence Generation

- Generation of sequences
  - Max no. of resonant swing-bys
  - Max no. of inward legs for an outer target
  - Max no. of outward legs for an inner target
- STOUR-like, energy based feasibility assessment
  - Circular, coplanar planet orbits
  - No phasing
  - No overturning of the relative velocity vector
- Hohmann Δv for sorting



Optimisation on the Feasible Set

- Let's call O_f the set of all the feasible trajectories.
- If the number of elements in O_f is small, apply systematically the pruning procedure to each one of them
- Otherwise use a global optimisation method for integer problems
- Evaluate each element of O_f by running the pruning+optimisation algorithm



University of Reading

UNIVERSITY of GLASGOW

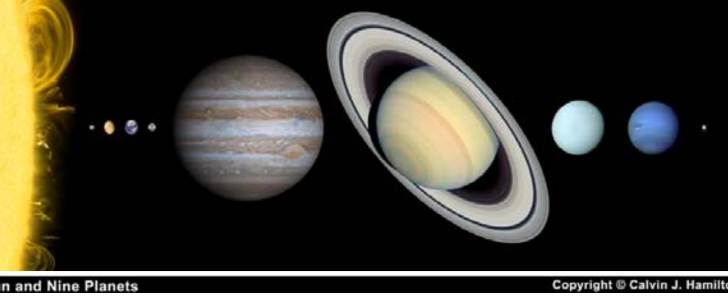
Example: GTOC 1

- Requirements
  - E to S transfer
  - Max 2 resonant swing-bys
  - Max 1 inward leg
  - Max v_{inf} at departure=3km/s
- Solution
  - 13 feasible sequences
  - Ranked according to the Hohmann total Δv (km/s)
  - The GTOC1 winning sequence is the 4th in the list
- Computational time to create the set O_f: **0.67s**

JPL solution sequence

Feasible sequences

1	E	V	E	E	M	S			28.14
2	E	V	E	E	M	M	S		28.14
3	E	E	V	E	E	M	S		28.14
4	E	V	E	E	E	J	S		28.19
5	E	V	E	E	J	J	S		28.19
6	E	E	V	E	E	J	S		28.19
7	E	V	E	E	M	J	S		28.19
8	E	V	E	M	M	J	S		29.49
9	E	V	E	E	M	M	J	S	29.49
10	E	V	E	E	M	J	J	S	29.49
11	E	V	E	M	M	J	J	S	29.49
12	E	E	V	E	E	M	J	S	29.49
13	E	E	V	E	M	M	J	S	29.49



Final Remarks

- Different models implies different sets of solution and a more or less difficult search. For example model 2 contains solutions that do not exist in model 1
- With model 1 the sum of the parts is not the whole therefore specific heuristics have to be used at each level
- About the algorithmic complexity for model 1 we can say that the number of function evaluations grows linearly with the number of levels
- At present we cannot say if the number of grid points required to have a sufficient characterisation of the search space for model 1 grows polynomially with the number of levels.
- Though both pruning procedures were proven to be robust when applied to the test cases, still the search for a solution is stochastic

University of Reading

UNIVERSITY of GLASGOW

Work in Progress

- Analysis of problem complexity
- Statistic tests on the search after pruning
- Alternative models and pruning procedures
- Iterative application of the pruning process can lead to a more reliable and effective pruning