Mine the Belt Challenge

Guillem Rodríguez Corominas[1], Albert López Serrano[1],
Roberto Maria Rosati[2], Mehmet Anil Akbay[1], Christian Blum[1]

(1) Artificial Intelligence Research institute (IIIA), CSIC, Bellaterra, Spain

(2) University of Udine, Udine, Italy

September 15, 2022

Outline

Outline for section 1

- 10.000 asteroids
- Limited fuel and time window
- No revisiting allowed
- To be mined: three kinds of materials $\{mat_1, mat_2, mat_3\}$ and fuel F

Objective:

Mine as much materials as possible, i.e. the objective function to be maximized is:

$$min_{i=1,2,3}(mat_i)$$

while complying with fuel and time restrictions.

- 10.000 asteroids
- Limited fuel and time window
- No revisiting allowed
- To be mined: three kinds of materials $\{mat_1, mat_2, mat_3\}$ and fuel F

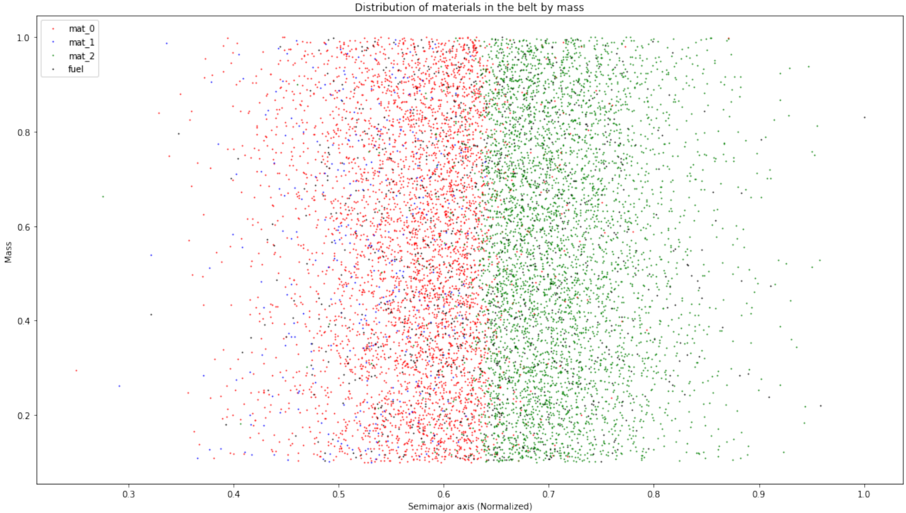
Objective:

Mine as much materials as possible, i.e. the objective function to be maximized is:

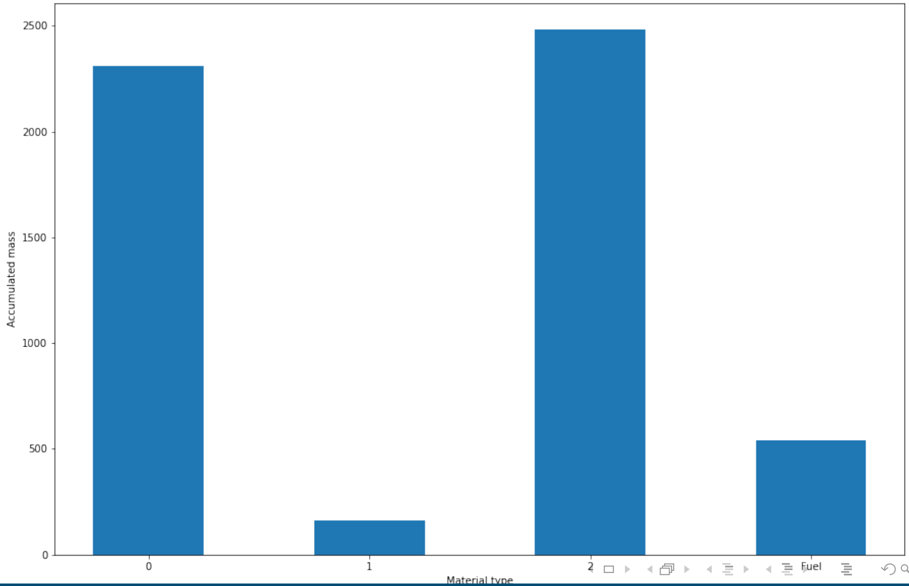
$$min_{i=1,2,3}(mat_i)$$

while complying with fuel and time restrictions.

Main Challenges: material distribution



Main Challenges: material amounts



Main Challenges

Key challenges

- Uneven material distribution and amounts
- Quick changing distances between asteroids
- Multiple orbital parameters governing motion $\Rightarrow$ hard to predict motion without explicitly calculating it

Hybrid Decision space

- Travel time and mining policy to be chosen in a continuous space
- Asteroid id's to be chosen from a shrinking discrete set

Main Challenges

Key challenges

- Uneven material distribution and amounts
- Quick changing distances between asteroids
- Multiple orbital parameters governing motion $\Rightarrow$ hard to predict motion without explicitly calculating it

Hybrid Decision space

- Travel time and mining policy to be chosen in a continuous space
- Asteroid id's to be chosen from a shrinking discrete set

Main Challenges

Key challenges

- Uneven material distribution and amounts
- Quick changing distances between asteroids
- Multiple orbital parameters governing motion $\Rightarrow$ hard to predict motion without explicitly calculating it

Hybrid Decision space

- Travel time and mining policy to be chosen in a continuous space
- Asteroid id's to be chosen from a shrinking discrete set

Main Challenges

Key challenges

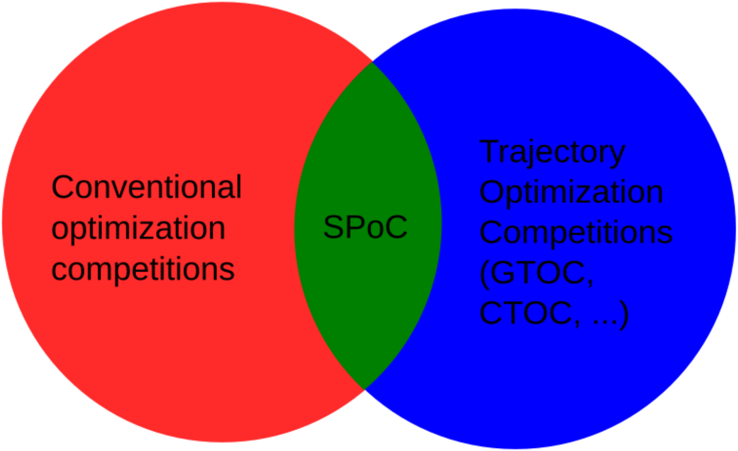
- Uneven material distribution and amounts
- Quick changing distances between asteroids
- Multiple orbital parameters governing motion $\Rightarrow$ hard to predict motion without explicitly calculating it

Hybrid Decision space

- Travel time and mining policy to be chosen in a continuous space
- Asteroid id's to be chosen from a shrinking discrete set

Outline for section 2

Similar competitions



Available tools

Combinatorial Optimization techniques

- Local Search
- Randomization
- Hybridization

Trajectory Optimization Techniques

- Gravitational Dynamics
- Homotopical techniques
- Single and multi-leg optimization

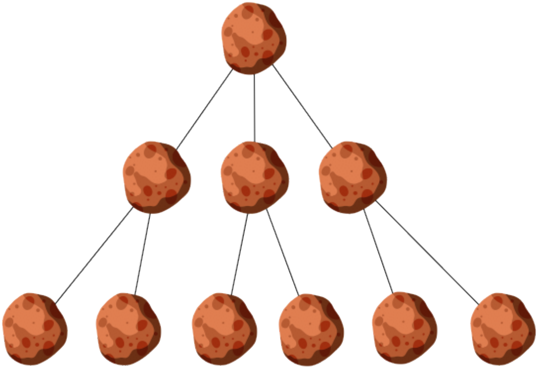
Common Techniques

- Greedy Algorithms
- Tree Search (Beamsearch)
- Deep Reinforcement Learning

Outline for section 3

Problem Modelling: Discrete Space

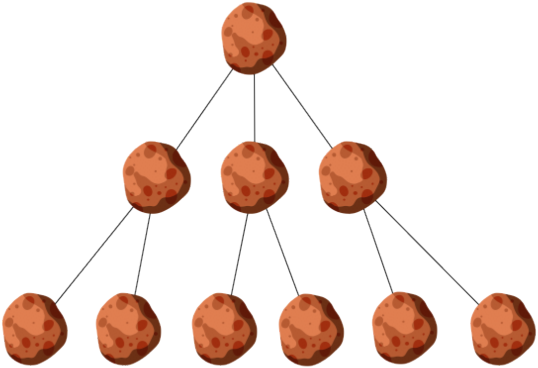
- Tree modeling of the combinatorial problem



Why? $\Rightarrow$ Order matters

Problem Modelling: Discrete Space

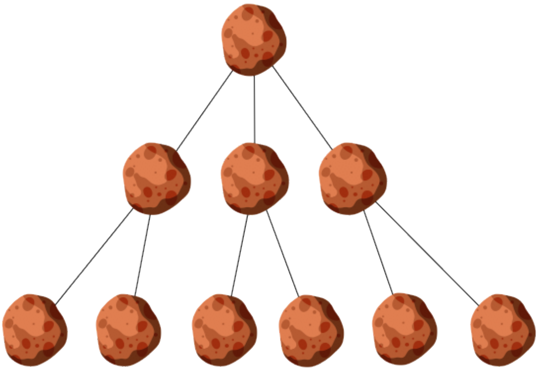
- Tree modeling of the combinatorial problem



Why? ⇒ Order matters

Problem Modelling: Discrete Space

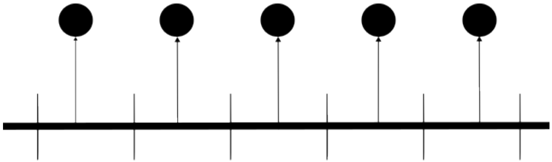
- Tree modeling of the combinatorial problem



Why? $\Rightarrow$ Order matters

Problem Modelling: Continuous Space

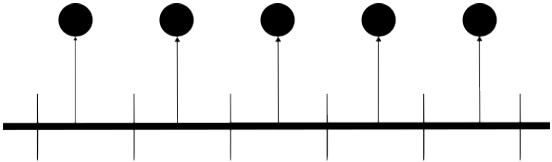
- Discretization of the continous variables



Why? ⇒ Simplicity

Problem Modelling: Continuous Space

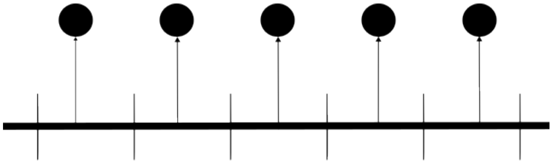
- Discretization of the continous variables



Why? ⇒ Simplicity

Problem Modelling: Continuous Space

- Discretization of the continous variables



Why? $\Rightarrow$ Simplicity

Solving Strategies

Id's selection

- **First approach:** Greedy depth-first strategy
- **Further development:** Beamsearch algorithm

$\Rightarrow$

Discrete parameter selection

- Mining time was fixed: $t_{mining} = asteroid\_mass \times 30$
- Flight time was chosen greedily based on ΔV and t_{flight} tradeoffs

Id's selection

- **First approach:** Greedy depth-first strategy
- **Further development:** Beamsearch algorithm

$\Rightarrow$

Discrete parameter selection

- Mining time was fixed: $t_{mining} = asteroid\_mass \times 30$
- Flight time was chosen greedily based on ΔV and t_{flight} tradeoffs

Solving Strategies

Id's selection

- **First approach:** Greedy depth-first strategy
- **Further development:** Beamsearch algorithm

$\Rightarrow$

Discrete parameter selection

- Mining time was fixed: $t_{mining} = asteroid\_mass \times 30$
- Flight time was chosen greedily based on ΔV and t_{flight} tradeoffs

Solving Strategies

Id's selection

- **First approach:** Greedy depth-first strategy
- **Further development:** Beamsearch algorithm

$\Rightarrow$

Discrete parameter selection

- Mining time was fixed: $t_{mining} = asteroid\_mass \times 30$
- Flight time was chosen greedily based on ΔV and t_{flight} tradeoffs

Solving Strategy

$$\begin{gathered}\textit{Current state understanding}\\ \downarrow \\ \textit{Decision regarding the next asteroid}(s)\\ \downarrow \\ t_{flight} \textit{ is decided}\\ \downarrow \\ \textit{Mining and state update}\end{gathered}$$

1. Possible times of flight $t_{flight} \in (0, 1827 - t_{current}]$ were discretized onto the set $t_{flight} \in \{5, 10, 15, 20, 25\}$
2. Lambert problem was solved by those t_{flight} values and ΔV were extracted
3. The flights complying with problem restrictions were evaluated under the punctuation:

$$\frac{1}{\Delta V \sqrt{t_{flight}}}$$

4. Best performing t_{flight} value (if any) was selected

t_{flight}

1. Possible times of flight $t_{flight} \in (0, 1827 - t_{current}]$ were discretized onto the set $t_{flight} \in \{5, 10, 15, 20, 25\}$
2. Lambert problem was solved by those t_{flight} values and ΔV were extracted
3. The flights complying with problem restrictions were evaluated under the punctuation:
$$\frac{1}{\Delta V \sqrt{t_{flight}}}$$
4. Best performing t_{flight} value (if any) was selected

t_{flight}

1. Possible times of flight $t_{flight} \in (0, 1827 - t_{current}]$ were discretized onto the set $t_{flight} \in \{5, 10, 15, 20, 25\}$
2. Lambert problem was solved by those t_{flight} values and ΔV were extracted
3. The flights complying with problem restrictions were evaluated under the punctuation:

$$\frac{1}{\Delta V \sqrt{t_{flight}}}$$

4. Best performing t_{flight} value (if any) was selected

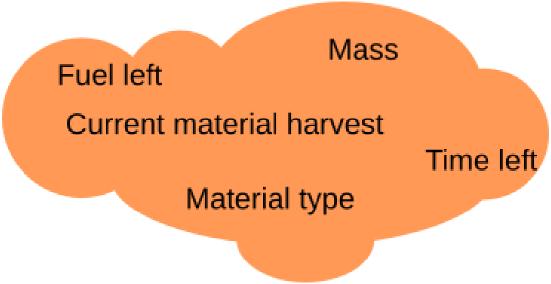
1. Possible times of flight $t_{flight} \in (0, 1827 - t_{current}]$ were discretized onto the set $t_{flight} \in \{5, 10, 15, 20, 25\}$
2. Lambert problem was solved by those t_{flight} values and ΔV were extracted
3. The flights complying with problem restrictions were evaluated under the punctuation:

$$\frac{1}{\Delta V \sqrt{t_{flight}}}$$

4. Best performing t_{flight} value (if any) was selected

Id Selection

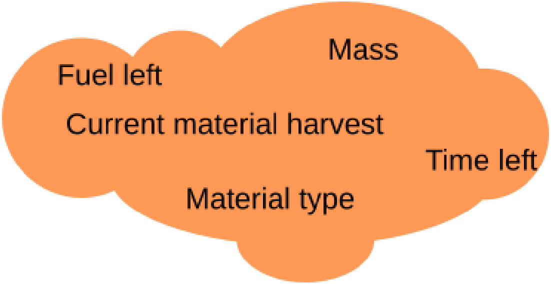
When deciding which would be the next asteroid in a chain, we had different parameters available to *score* its potential:



and different ways to relate them.

Id Selection

When deciding which would be the next asteroid in a chain, we had different parameters available to *score* its potential:



and different ways to relate them.

Id Selection

After many trials, we opted to evaluate the *potential value* of an asteroid of material type $i \in \{0, 1, 2\}$ and mass m as :

$$\frac{m(\sum_{k \neq i} mat_k + 0.5)}{(\sum_{j=0}^{2} mat_j + 1)\Delta V \sqrt{t_{flight}}}$$

and for fuel asteroids:

$$\frac{-m \cdot log(fuel)}{\Delta V \sqrt{t_{flight}}}$$

Id Selection

After many trials, we opted to evaluate the *potential value* of an asteroid of material type $i \in \{0, 1, 2\}$ and mass m as :

$$\frac{m(\sum_{k \neq i} mat_k + 0.5)}{(\sum_{j=0}^{2} mat_j + 1)\Delta V \sqrt{t_{flight}}}$$

and for fuel asteroids:

$$\frac{-m \cdot log(fuel)}{\Delta V \sqrt{t_{flight}}}$$

Id Selection

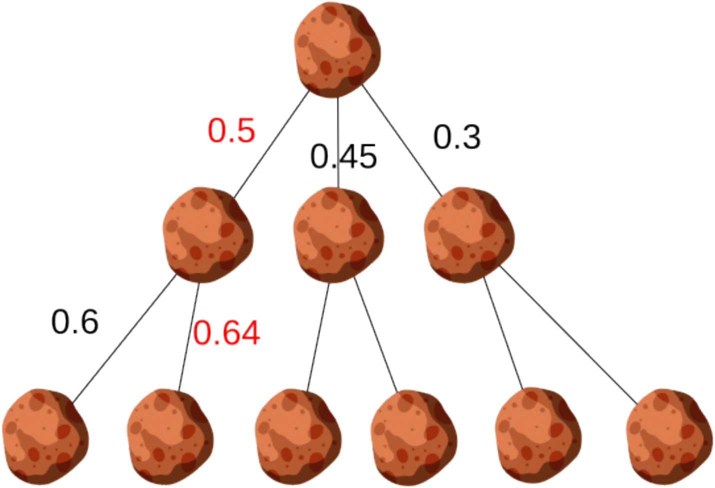


Figure: Greedy tree construction

Id Selection

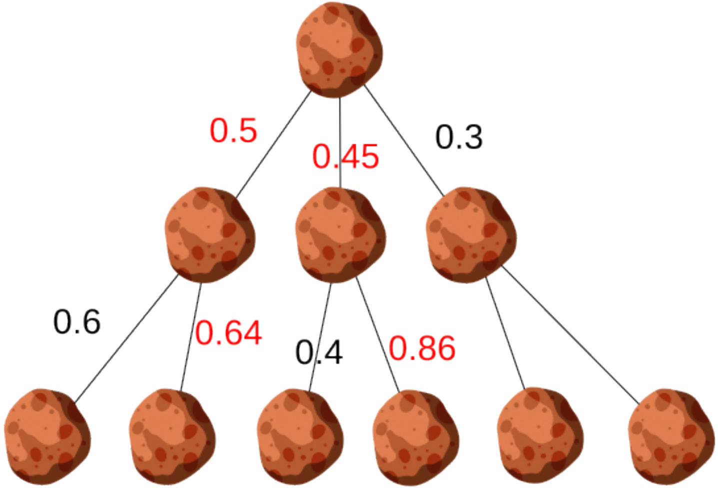


Figure: Beamsearch-based tree construction

Id Selection

Beamsearch-based tree construction

1. Have a set of *partial solutions*, or an empty one.
2. Propagate the tree with the most k promising sons of each node in *partial solutions*.
3. Store the complete solutions, if any
4. Prune the new set of *partial solutions* to the best n partial solutions

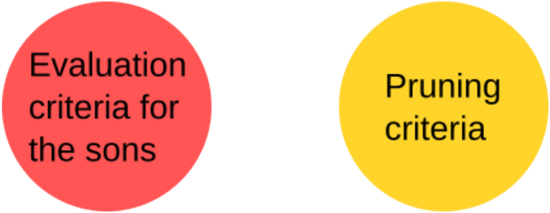


Figure: Main decisions in beamsearch

Pruning criteria

After reviewing different possibilities, a greedy approach using the following formula was used to decide which partial solutions were kept:

$$\frac{\sum_{i=0}^{2} mat_i + fuel - \sigma(mat) * \frac{t}{1827}}{t + t_{wasted}}$$

where t_{wasted} is the total time that could be spent mining fuel in asteroids that filled the fuel gas instead.

Outline for section 4

Parameter tuning

For the experimental evaluation of the algorithm the following decisions were made:

- All 10.000 asteroids were considered as seeds, and the algorithm was run on them
- Beamsearch parameters were hand-tuned:
  - Only the closest 500 asteroids (orbital distance) to the current one were considered at each step
  - We kept the 20 best scoring *sons* of each partial solution
  - At each time step we kept at most 250 partial solutions

Parameter tuning

For the experimental evaluation of the algorithm the following decisions were made:

- All 10.000 asteroids were considered as seeds, and the algorithm was run on them
- Beamsearch parameters were hand-tuned:
  - Only the closest 500 asteroids (orbital distance) to the current one were considered at each step
  - We kept the 20 best scoring *sons* of each partial solution
  - At each time step we kept at most 250 partial solutions

Parameter tuning

For the experimental evaluation of the algorithm the following decisions were made:

- All 10.000 asteroids were considered as seeds, and the algorithm was run on them
- Beamsearch parameters were hand-tuned:
  - Only the closest 500 asteroids (orbital distance) to the current one were considered at each step
  - We kept the 20 best scoring *sons* of each partial solution
  - At each time step we kept at most 250 partial solutions

Parameter tuning

For the experimental evaluation of the algorithm the following decisions were made:

- All 10.000 asteroids were considered as seeds, and the algorithm was run on them
- Beamsearch parameters were hand-tuned:
  - Only the closest 500 asteroids (orbital distance) to the current one were considered at each step
  - We kept the 20 best scoring *sons* of each partial solution
  - At each time step we kept at most 250 partial solutions

Results

- The greedy algorithm achieved results around 3 and 7, depending on the seed and specific function used
- The beamsearch algorithm achieved results between 9 and 10.103

Outline for section 5

Future Work

Improvements on different steps of the algorithm could and should be done to further explore the problem:

Single leg optimization

The t_{flight} could be optimized searching for the best approximation of the Pareto frontier for the ΔV and t_{flight}

Greedy criteria

Both pruning and tree construction criteria could be improved using **Deep-Reinforcement Learning**

Id selection

- Multi-step explorations could be useful to evaluate the potential of candidate asteroids
- Data-driven trajectory planning could improve the predictions by accounting for orbital elements and global planning

Future Work

Improvements on different steps of the algorithm could and should be done to further explore the problem:

Single leg optimization

The t_{flight} could be optimized searching for the best approximation of the Pareto frontier for the ΔV and t_{flight}

Greedy criteria

Both pruning and tree construction criteria could be improved using **Deep-Reinforcement Learning**

Id selection

- Multi-step explorations could be useful to evaluate the potential of candidate asteroids
- Data-driven trajectory planning could improve the predictions by accounting for orbital elements and global planning

Future Work

Improvements on different steps of the algorithm could and should be done to further explore the problem:

Single leg optimization

The t_{flight} could be optimized searching for the best approximation of the Pareto frontier for the ΔV and t_{flight}

Greedy criteria

Both pruning and tree construction criteria could be improved using **Deep-Reinforcement Learning**

Id selection

- Multi-step explorations could be useful to evaluate the potential of candidate asteroids
- Data-driven trajectory planning could improve the predictions by accounting for orbital elements and global planning

Future Work

Improvements on different steps of the algorithm could and should be done to further explore the problem:

Single leg optimization

The t_{flight} could be optimized searching for the best approximation of the Pareto frontier for the ΔV and t_{flight}

Greedy criteria

Both pruning and tree construction criteria could be improved using **Deep-Reinforcement Learning**

Id selection

- Multi-step explorations could be useful to evaluate the potential of candidate asteroids
- Data-driven trajectory planning could improve the predictions by accounting for orbital elements and global planning

Future work

The performance of the beamsearch in itself could be improved in different ways:

Beamsearch extensions

- Probabilistic tree building
- Population Based Beam-ACO
- Multi-greedy scoring for pruning
- Changing the greedy mechanisms for more complex approaches

Solution refining

- Backtracking
- Random perturbations and reconstruction of solutions

Future work

The performance of the beamsearch in itself could be improved in different ways:

Beamsearch extensions

- Probabilistic tree building
- Population Based Beam-ACO
- Multi-greedy scoring for pruning
- Changing the greedy mechanisms for more complex approaches

Solution refining

- Backtracking
- Random perturbations and reconstruction of solutions

Future work

The performance of the beamsearch in itself could be improved in different ways:

Beamsearch extensions

- Probabilistic tree building
- Population Based Beam-ACO
- Multi-greedy scoring for pruning
- Changing the greedy mechanisms for more complex approaches

Solution refining

- Backtracking
- Random perturbations and reconstruction of solutions

Questions?