The Multi-Inter-Distance Constraint

Pierre Ouellet and Claude-Guy Quimper
Université Laval, Québec, Canada
pierre.ouellet.4@ulaval.ca, claude-guy.quimper@ift.ulaval.ca

Abstract

We introduce the MULTI-INTER-DISTANCE constraint that ensures no more than m variables are assigned to values lying in a window of p consecutive values. This constraint is useful for modelling scheduling problems where tasks of processing time p compete for m identical resources. We present a propagator that achieves bounds consistency in cubic time. Experiments show that this new constraint offers a much stronger filtering than an edge-finder and that it allows to solve larger instances of the runway scheduling problem.

The Constraint

The constraint MULTI-INTER-DISTANCE is satisfied when no more than m variables are assigned within a window of p consecutive values.

$$\textsc{Multi-Inter-Distance}([X_1,\dots,X_n],m,p)$$
$$\Longleftrightarrow$$
$$\forall v \; |\{i \mid X_i \in [v, v+p)\}| \le m$$
$$\Longleftrightarrow$$
$$\forall i \; |\{j \mid 0 \le X_j - X_i \le p\}| \le m$$

This constraint models a scheduling problem where n tasks with processing time p compete for m identical resource.

Example

Consider the constraint over 5 variables.

MULTI-INTER-DISTANCE($[X_1, ..., X_5]$, $m = 2$, $p = 3$)

- dom(X_1) = [7, 9)
- dom(X_2) = [2, 4)
- dom(X_3) = [4, 7)
- dom(X_4) = [2, 7)
- dom(X_5) = [3, 5)

A solution to the constraint is the assignment

$$X_1 = 8, X_2 = 2, X_3 = 5, X_4 = 6, X_5 = 3.$$

After enforcing bounds consistency, we obtain these new domains:

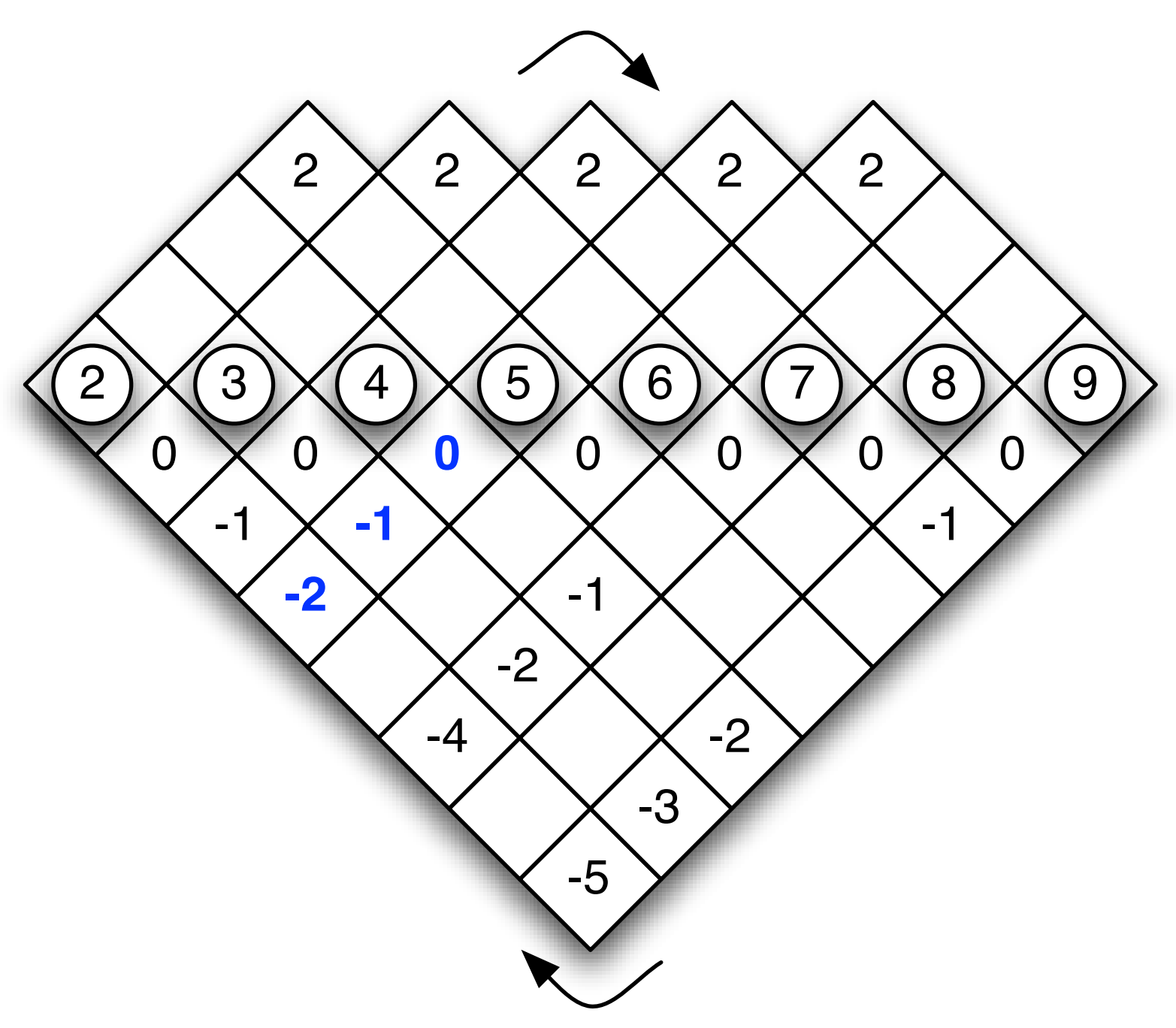
- dom(X_1) = [8, 9)
- dom(X_2) = [2, 3)
- dom(X_3) = [5, 7)
- dom(X_4) = [5, 7)
- dom(X_5) = [3, 4)

The Scheduling Graph

The scheduling graph is a tool to decide whether the constraint is satisfiable. The scheduling graph has the following nodes and edges.

- There is a **node** for each time point.
- There is a **forward edge** (a, a + p) with weight m for each time point a.
- There is a **null edge** (a, a - 1) with null weight for each time point a.
- There is a **backward edge** (b, a) with a negative weight whose absolute value is the number of domains contained in the interval [a, b).

Example of a Scheduling Graph



Properties of the Scheduling Graph

- **Theorem**: The MULTI-INTER-DISTANCE constraint is satisfiable if and only if the scheduling graph has no negative cycles.
- **Definition:** The **altered graph** G_i^v is the scheduling graph of the original problem where the upper bound of the domain of X_i is set to v. For instance, the graph G_3^5 is obtained by decrementing the weights in blue in the graph above by one.
- **Corollary**: If the altered graph G_i^v has a negative cycle then $X_i \ge v$.
- **Theorem**: If $X_i = v$ has no interval support and the domain of X_j has a larger upper bound than the domain of X_i, then $X_j = v$ has no interval support.
- **Theorem**: Let u^* be the smallest domain upper bound that is larger than $\min(\mathrm{dom}(X_i))$. Let t be the largest value s.t. the shortest path in $G_i^{u^*}$ from $\min(\mathrm{dom}(X_i))$ to t is null, then $X_i = t$ is the smallest interval support for X_i.

The Filtering Algorithm

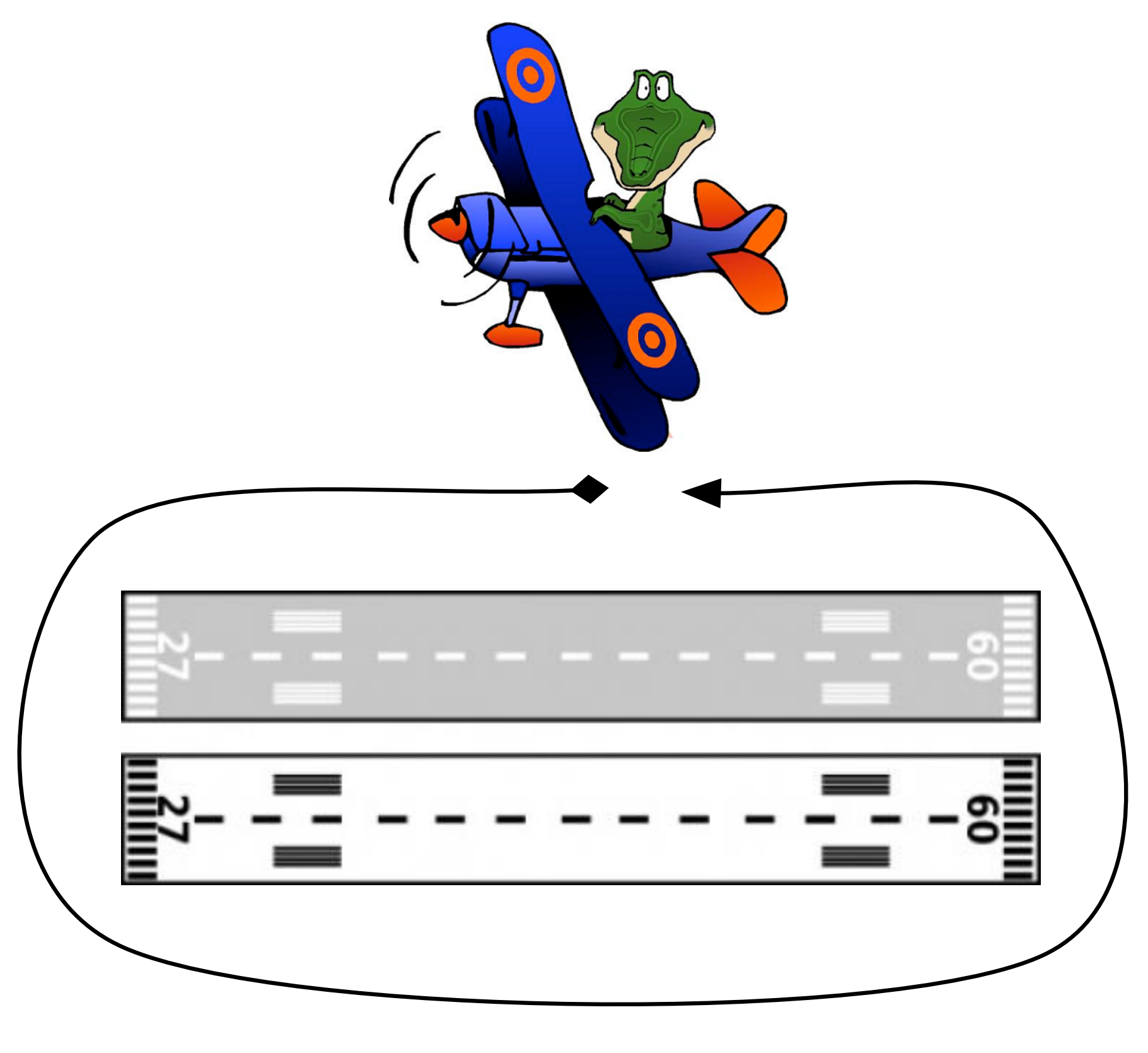
```
Algorithm 1: PruneLowerBounds($[X_1,\dots,X_n]$)
Sort the variables in non-decreasing domain upper bounds
Let $l_i = \min(\mathrm{dom}(X_i)$ and $u_i = \max(\mathrm{dom}(X_i))$
$U \leftarrow \{u_1,\dots,u_n\}$
$F \leftarrow \emptyset$
for $i \in 1..n$ do
    repeat
        $l_i \leftarrow \min([l_i, u_i) \setminus F)$
        $u^* \leftarrow \min((l_i, u_i] \cap U)$
        Compute the shortest distances from the node $l_i$ to
        all the other nodes $t$ in the altered scheduling graph $G_i^{u^*}$.
        if there is a negative cycle in $G_i^{u^*}$ then
            $F \leftarrow F \cup [l_i, u^*)$
    until there is no negative cycles in $G_i^{u^*}$
    $l_i \leftarrow \max(\{t \mid$ distance from $l_i$ to $t$ in $G_i^{u^*}$ is null$\})$
```

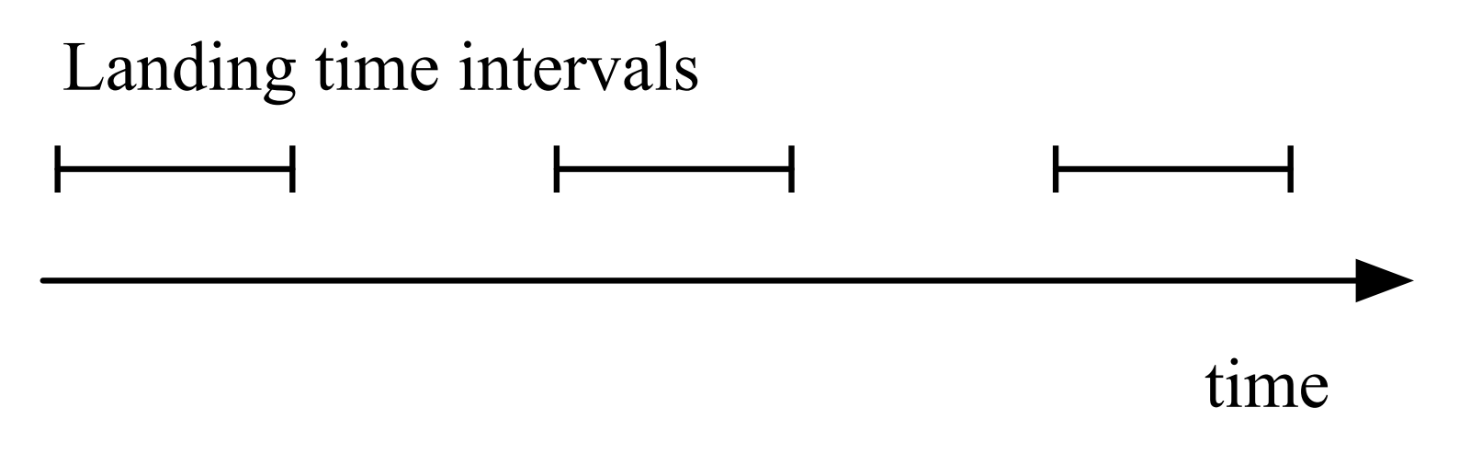
1. Process the domains in non-decreasing order of upper bound:
   1.1. Shrink the lower bound of the current domain according to the discovered forbidden regions.
   1.2. Find the smallest upper bound u* that is larger than the current domain lower bound l_i.
   1.3. If the altered scheduling graph $G_i^{u^*}$ has a negative cycle, create the new forbidden region $[l_i, u^*)$ and go back to 1.1.
   1.4. Find the largest value t whose distance from l_i is null and set t as the new lower bound of the current domain.

Complexity: $O(n^3 \min(1, \frac{p}{m}))$.

The Runway Scheduling Problem

In this problem, n planes need to land in an airport with m runways. The planes fly around the airport while waiting for their authorization to land. A plane can only land within specific time intervals in which it is in position for landing. Outside these time intervals, the plane cannot land. The goal is to dispatch the planes on the m runways while maximizing the minimum time between two successive landings.



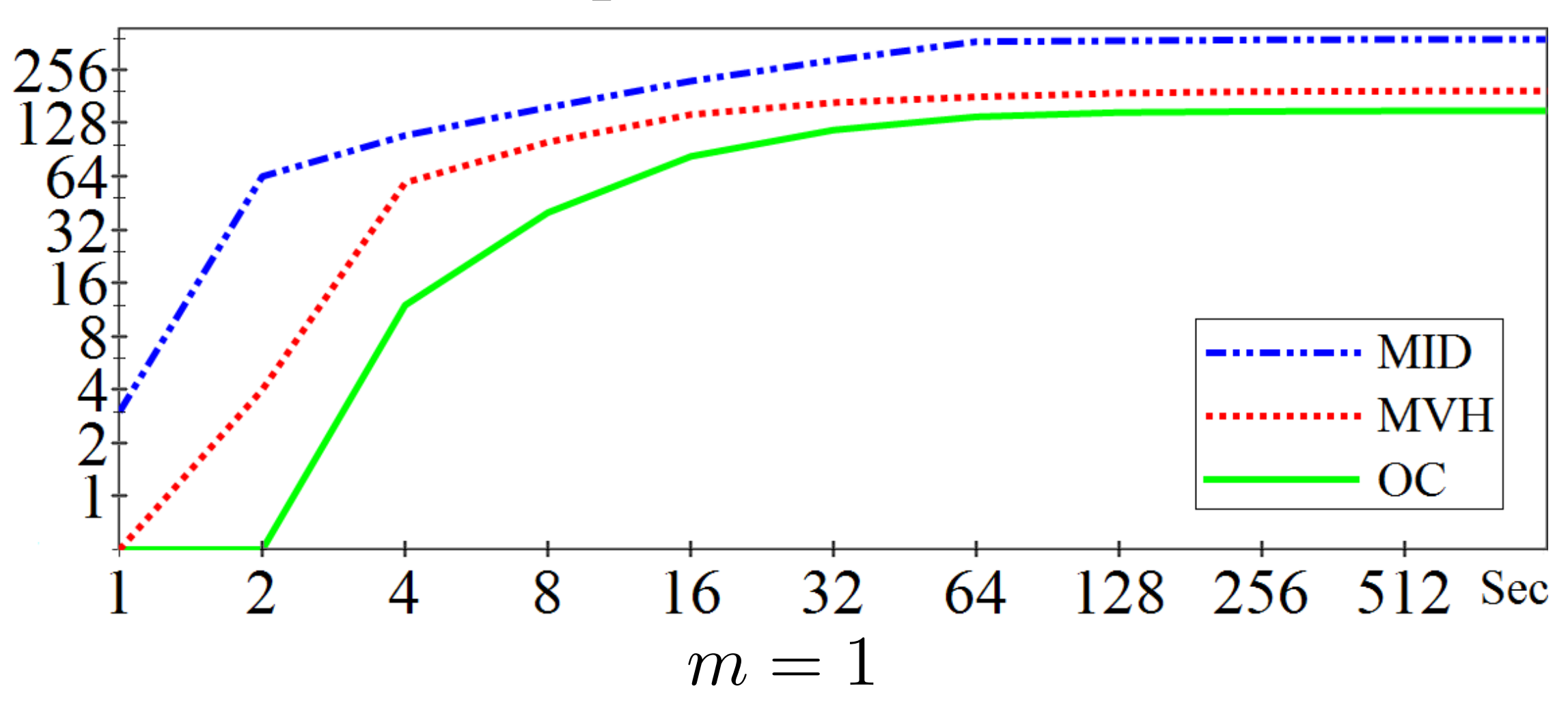


Summary:

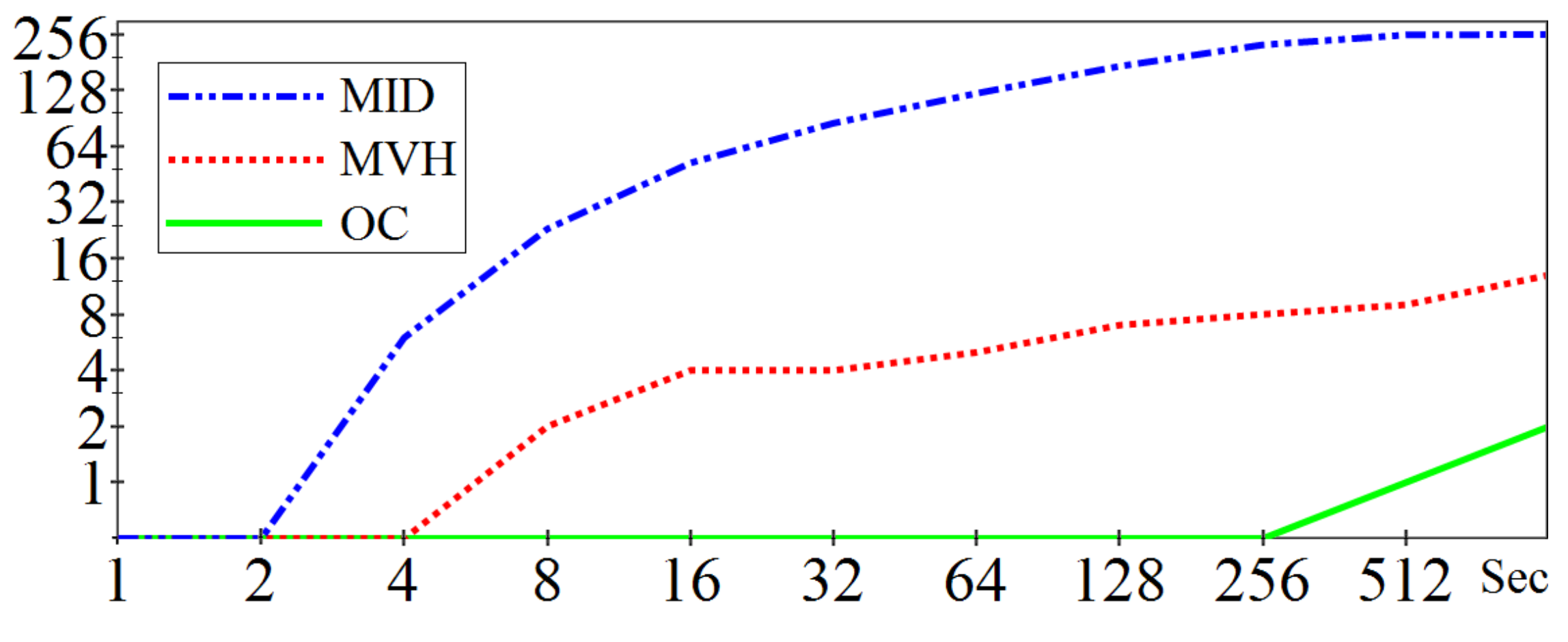
- n planes
- m runways
- Each plane has a set of time intervals within which landing is possible.
- The goal is to maximize the time between each landing on a same runway.

Experimental Results

Number of problems solved vs time



$m = 1$



$m \in \{2, 3\}$

MID: MULTI-INTER-DISTANCE MVH: Edge Finder
OC: Overload checking

Conclusion

Enforcing bounds consistency on the MULTI-INTER-DISTANCE constraint is done in cubic time. This new constraint filters more than the edge finder and the overload checking. Experiments show that bounds consistency leads to better computational times.