

Six Ways of integrating Symmetries within Non-Overlapping Constraints

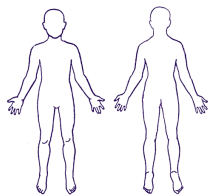
M. Ågren, N. Beldiceanu, M. Carlsson,
M. Sbihi, C. Truchet, S. Zampelli

Swedish Institute of Computer Science
École des Mines de Nantes
Université de Nantes



June 3, 2009

Outline



Background
Sweep-Based Filtering
Filtering *lex-chain* \wedge *non-overlapping*
Filtering *lex-chain* \wedge *cumulative*
Final Remarks

The *non-overlapping* Constraint

non-overlapping(\mathcal{O})

\mathcal{O} A set of k -dimensional orthotopes (hyper-rectangles) o_i with *origin* $o_i.x[d] \in \mathbb{Z}^k$, $1 \leq d \leq k$ and *size* $o_i.l[d] \in \mathbb{Z}_+^k$, $1 \leq d \leq k$.

- ▶ An orthotope is *monomorphic* if its size is fixed; otherwise, it is *polymorphic*.
- ▶ The constraint holds if there is no pair of orthotopes that forms a non-empty intersection.

The *lex-chain* Constraint

Popular Symmetry Breaking Constraint

$$(x_{11}, \dots, x_{1k}) \leq_{\text{lex}} \dots \leq_{\text{lex}} (x_{n1}, \dots, x_{nk})$$

Two-phase filtering algorithm (previous work)

1. For each $i \in [1, n]$, compute feasible lower bound L_i and upper bound U_i .
2. For each $i \in [1, n]$, filter $L_i \leq_{\text{lex}} (x_{i1}, \dots, x_{ik}) \leq_{\text{lex}} U_i$.

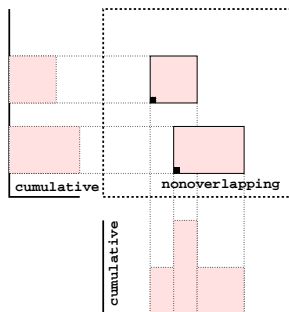
The *cumulative* Constraint

Important Redundant Constraint for *non-overlapping*

$$cumulative(\mathcal{T}, C)$$

\mathcal{T} A set of tasks with *origin*, *duration*, *end* and *height*.

$C \in \mathbb{Z}^+$ A *capacity* that cannot be exceeded at any time.



The *geost* Constraint

Combines *non-overlapping*, *lex-chain*, *cumulative*



$geost(k, \mathcal{O}, \mathcal{S}, \mathcal{R})$

k Number of **dimensions**.

\mathcal{O} Set of **objects** o with unique *object id* $o.oid$ (an integer), *shape id* $o.sid$, *origin* $o.x[d]$, $1 \leq d \leq k$, *optionally more* integer attributes.

\mathcal{S} Set of **shifted boxes** s with *shape id* $s.sid$, *shift offset* $s.t[d]$, $1 \leq d \leq k$, *size* $s.l[d]$ ($s.l[d] > 0$, $1 \leq d \leq k$), *optionally more* attributes (all integers).

\mathcal{R} Set of **rules**, referring to the above attributes.

Semantics Ground instance is true iff it satisfies the rules.

Sweep-Based Filtering

Forbidden Regions

The Sweep Idea

A generic, lightweight pruning approach to aggregating multiple constraints sharing some variables $[x_1, \dots, x_k]$.

Requirement: a way to compute *forbidden regions* for $[x_1, \dots, x_k]$.

Forbidden region for $[x_1, \dots, x_k]$ wrt. formula F

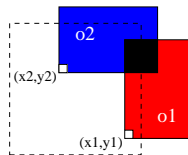
A nonempty k -dimensional orthotope r such that if $[x_1, \dots, x_k]$ is inside r , F is necessarily false.

Let $\mathcal{F}([x_1, \dots, x_k])$ denote the set of forbidden regions for $[x_1, \dots, x_k]$.

Sweep-Based Filtering

Computing $\mathcal{F}(o_1)$ wrt. $non-overlapping(\{o_1, o_2\})$

Let o_1, o_2 be rectangles with origin $(x_1, y_1) \in \mathbb{Z}^2, (x_2, y_2) \in \mathbb{Z}^2$ and of size $(w_1, h_1) \in \mathbb{Z}_+^2, (w_2, h_2) \in \mathbb{Z}_+^2$. We have:



$$\begin{aligned} \neg non-overlapping(\{o_1, o_2\}) &\Leftrightarrow \\ x_1 + w_1 > x_2 \wedge x_2 + w_2 > x_1 \wedge y_1 + h_1 > y_2 \wedge y_2 + h_2 > y_1 &\Leftrightarrow \\ (x_1, y_1) \in \{(x, y) \in \mathbb{Z}^2 \mid x > x_2 - w_1\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid x < x_2 + w_2\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid y > y_2 - h_1\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid y < y_2 + h_2\} & \end{aligned}$$

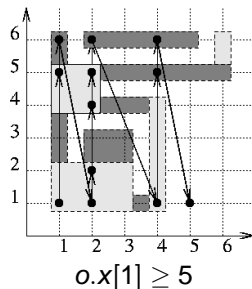
If o_2 is not fixed, we take the intersection of all forbidden regions:

$$\begin{aligned} (x_1, y_1) \in \{(x, y) \in \mathbb{Z}^2 \mid x > \overline{x_2} - w_1\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid x < \underline{x_2} + w_2\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid y > \overline{y_2} - h_1\} & \\ \cap \{(x, y) \in \mathbb{Z}^2 \mid y < \underline{y_2} + h_2\} & \end{aligned}$$

Sweep-Based Filtering

Lexicographic Sweep for $C \equiv non-overlapping(\{o_1, \dots, o_n\})$

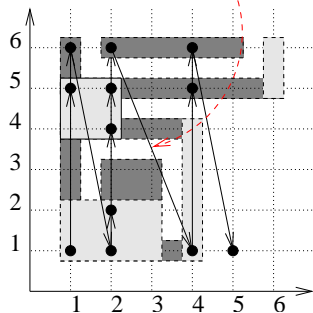
- ▶ To adjust the lower bound of $o_i.x[1]$:
 1. Compute $\mathcal{F}(o_i)$ wrt. C . $O(kn)$ cost.
 2. Seek the *lexicographically smallest position* (x', y') that is outside $\mathcal{F}(o_i)$, a.k.a. *witness* (o_i) . $O(kn)$ cost.
 3. Impose $o_i.x[1] \geq x'$.
- ▶ Similarly for upper bounds, all k dimensions, all n orthotopes, until fixpoint. $O(k^2 n^2)$ cost.



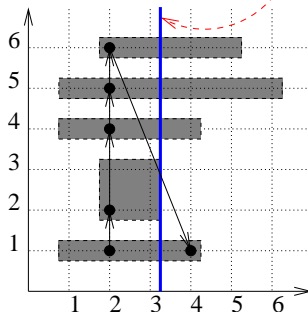
Sweep-Based Filtering

Lexicographic Sweep for $C \equiv \text{non-overlapping}(\mathcal{O})$

Why this jump?



Minimum extent in the MSD during one pass in the LSD.



Sweep-Based Filtering

Domination 1: Incremental Sweep

Filtering *non-overlapping*($\{o_1, \dots, o_n\}$)

if we have computed *witness*(o_i)

and o_i dominates o_j

then computing *witness*(o_j) can start at *witness*(o_i).

o_i dominates o_j iff

1. $\text{dom}(o_j.x[d]) \subseteq \text{dom}(o_i.x[d]), 1 \leq d \leq k,$
2. $\text{dom}(o_j.l[d]) \subseteq \text{dom}(o_i.l[d]), 1 \leq d \leq k,$

Sweep-Based Filtering

Domination 2: Incremental Forbidden Regions

Filtering *non-overlapping*($\{o_1, \dots, o_n\}$)

If we have computed *witness*(o_i)

and o_i strongly dominates o_{i+1}

then computing *witness*(o_{i+1}) can start at *witness*(o_i)

and $\mathcal{F}(o_{i+1})$ can be incrementally computed from $\mathcal{F}(o_i)$.

o_i strongly dominates o_j iff

1. $\text{dom}(o_j.x[d]) = \text{dom}(o_i.x[d]), 1 \leq d \leq k,$
2. $\text{dom}(o_j.l[d]) = \text{dom}(o_i.l[d]), 1 \leq d \leq k,$

This brings the complexity down from $O(k^2 n^2)$ to $O(k^2 n)$.

Sweep-Based Filtering

Lexicographic Sweep for $o_1 \leq_{\text{lex}} \dots \leq_{\text{lex}} o_n$

Two-phase filtering algorithm (previous work)

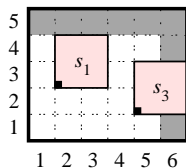
1. For each $i \in [1, n]$, compute feasible lower bound L_i and upper bound U_i .
2. For each $i \in [1, n]$, filter o_i wrt. $L_i \leq_{\text{lex}} o_i \leq_{\text{lex}} U_i$.

Two-phase filtering algorithm (adapted to sweep)

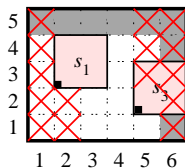
1. For each $i \in [1, n]$, compute feasible lower bound L_i and upper bound U_i .
2. For each $i \in [1, n]$, compute $\mathcal{F}(o_i)$ wrt. L_i and U_i .

Sweep-Based Filtering

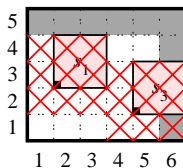
Aggregating $\text{non-overlapping}(\{s_1, s_2, s_3\})$ and $s_1 \leq_{\text{lex}} s_2 \leq_{\text{lex}} s_3$



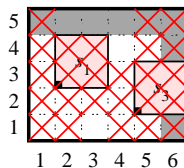
(A)



(B)



(C)



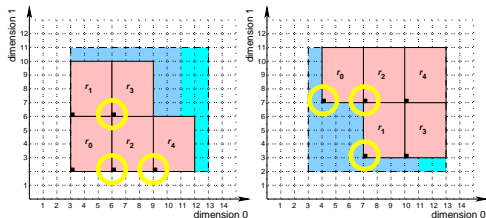
(D)

- (B) Forbidden points wrt. lex-chain
- (C) Forbidden points wrt. non-overlapping
- (D) Aggregated forbidden points.

Combining $lex-chain \wedge non-overlapping$

Monomorphic Case—All Orthotopes Have the Same Shape

1. Simulate “leftflush” greedy placement.
2. Simulate “rightflush” greedy placement.
3. Obtain lexicographic bounds \Rightarrow forbidden points.



- ▶ Placing five rectangles of size 3×4 .
- ▶ **N.B.** Neither $lex-chain$ nor $non-overlapping$ will propagate in isolation.

Combining $lex-chain \wedge non-overlapping$

Polymorphic Case—All Orthotopes Have the Same Multiset of Sizes

Typical case: orthotope can be rotated, e.g. on a pallet.

Bound 1 Reduce all sizes to the smallest value ℓ_{\min} . Apply monomorphic case.

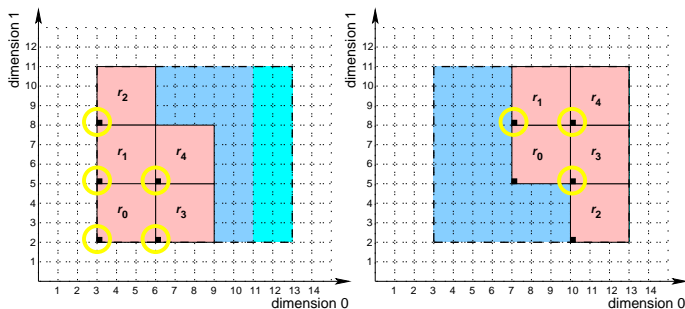
Bound 2 Split all boxes into identical cubes of size ℓ , $1 \leq \ell \leq \ell_{\min}$, possibly with some loss. Then:

- ▶ Simulate “leftflush” greedy placement of the cubes.
- ▶ Simulate “rightflush” greedy placement of the cubes.
- ▶ Obtain lexicographic bounds *with loss compensation* \Rightarrow forbidden points.

Combining $\text{lex-chain} \wedge \text{non-overlapping}$

Polymorphic Case—Bound 1

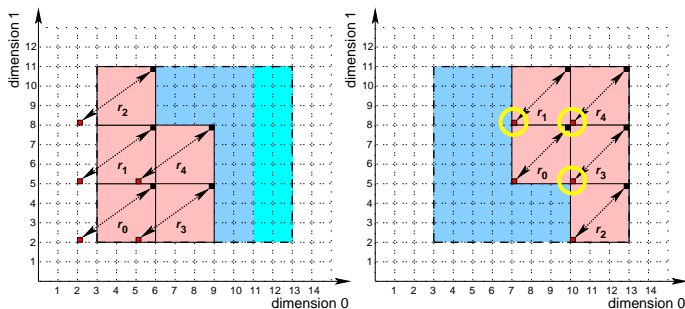
Placing five rectangles of size 3×4 or 4×3 .



Combining $\text{lex-chain} \wedge \text{non-overlapping}$

Polymorphic Case—Bound 2, $\ell = 3$

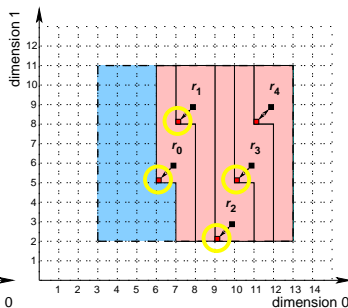
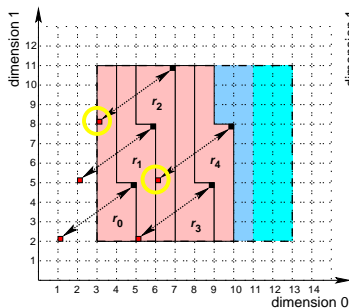
Placing five rectangles of size 3×4 or 4×3 .



Combining $\text{lex-chain} \wedge \text{non-overlapping}$

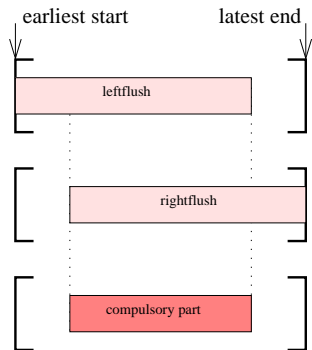
Polymorphic Case—Bound 2, $\ell = 1$

Placing five rectangles of size 3×4 or 4×3 .

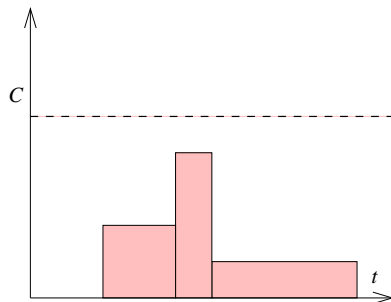


Combining $lex-chain \wedge cumulative$

Compulsory Part Profile



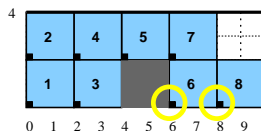
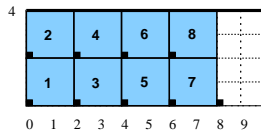
Profile of aggregated compulsory parts



Combining $\text{lex-chain} \wedge \text{cumulative}$

Considering the Compulsory Part Profile

Given a chain $t_1 \preceq \dots \preceq t_8$ implied by lex-chain



- ▶ Some other task has a compulsory part between $t = 4$ and $t = 6$.
- ▶ Simulate “leftflush” greedy placement to get lower bounds.
- ▶ Simulate “rightflush” greedy placement to get upper bounds.

Min Energy Filtering in *cumulative*

Let:

free the amount of free space on top of an interval I

σ the slack of I (max allowed unused space)

$max_overlap(S)$ the sum of the max overlaps of the tasks S

Pruning rule assuming that $t \in \mathcal{T}$ does not overlap I

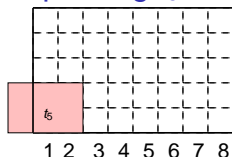
$$free - (max_overlap(\mathcal{T}) - max_overlap(\{t\})) > \sigma$$

\Rightarrow

$$overlap(t) \geq free - (max_overlap(\mathcal{T}) - max_overlap(\{t\})) - \sigma$$

Tasks t_1, \dots, t_5 of height 2 and duration 3, pruning t_5

- ▶ $I = 1..8$, $free = 40$, $\sigma = 13$,
 $max_overlap(\{t_1, \dots, t_5\}) = 30$
- ▶ $40 - (30 - 6) = 16 > 13 \checkmark$
- ▶ $\Rightarrow overlap(t_5) \geq 40 - (30 - 6) - 13 = 3$



Min Energy Filtering in $cumulative \wedge lex-chain$

Given a chain $t_1 \preceq \dots \preceq t_i$ implied by $lex-chain$

If t_i ends before l , each of t_1, \dots, t_{i-1} also ends before l .

Pruning rule assuming that $t_i \in \mathcal{T}$ ends before l

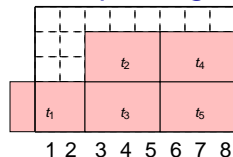
$$free - (max_overlap(\mathcal{T}) - max_overlap(\{t_1, \dots, t_i\})) > \sigma$$

$$\Rightarrow$$

$$overlap(\{t_1, \dots, t_i\}) \geq free - (max_overlap(\mathcal{T}) - max_overlap(\{t_1, \dots, t_i\})) - \sigma$$

Chain $t_1 \preceq \dots \preceq t_5$ of height 2 and duration 3, pruning t_5

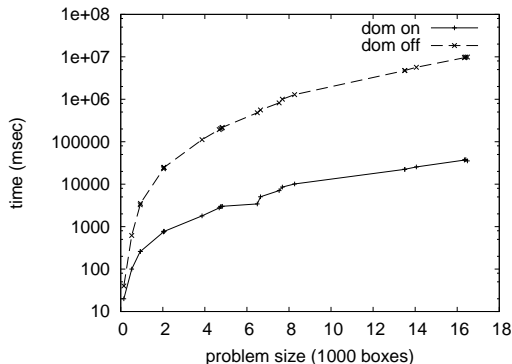
- ▶ $l = 1..8$, $free = 40$, $\sigma = 13$,
 $max_overlap(\{t_1, \dots, t_5\}) = 30$
- ▶ $40 - (30 - 30) = 40 > 13$ ✓
- ▶ $\Rightarrow overlap(\{t_1, \dots, t_5\}) \geq 40 - (30 - 30) - 13 = 27$



Performance Evaluation of Strong Domination

A Real-World 3D Bin-Packing Problem

- ▶ Items of 59 different shapes and weights.
- ▶ Max weight per container.
- ▶ Some items must be on the floor, others on top of piles.
- ▶ Modeled as a single 6D *geost* constraint running in *greedy* mode.



Performance Summary

Integrating *lex-chain* \wedge (*cumulative* resp. *non-overlapping*)

- ▶ **(Strong) domination** brings the complexity of filtering n k -dimensional orthotopes down from $O(k^2 n^2)$ to $O(k^2 n)$.
- ▶ The integration is quite effective in terms of search effort.
- ▶ The integration is sometimes faster and sometimes slower than a separate *lex-chain*.
- ▶ For monomorphic instances, integration into *cumulative* is the more effective.
- ▶ For polymorphic instances, integration into *non-overlapping* is the more effective.
- ▶ Integration into *cumulative* is the least expensive.

Conclusion

