Generalised Spatial and Temporal Placement Constraint: Current Status and Evolution

N. Beldiceanu,
École des Mines de Nantes, LINA UMR 6241 CNRS, FR-44307, France
Nicolas.Beldiceanu@emn.fr

With contributions of: M. Ågreen, M. Carlsson, (SICS)
S. Demassey, E. Poder, R. Sadek,
M. Sbihi, C. Truchet, S. Zampelli (LINA)

EuroControl déc. 2008
CONTEXT

We present a **spatio-temporal** global constraint kernel *geost*. Initially developed in the ongoing EU project NetWMS (by **LINA** and **SICS**). Initial focus was the geometrical aspect (*even if time was introduced*).

Extended in some project starting in 2009 (SelfXL).

WHAT IS IT ALL ABOUT ?

*Dynamic problems which mix geometrical and temporal aspects.*
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

geost on the web
A Generic Placement Kernel: geost
A Generic Placement Kernel: geost

geost(2,

[object(1,1,[1,2], 2,12,14),
 object(2,5,[2,1],10,12,22),
 object(3,8,[4,1],10,12,22),
 object(4,9,[1,1],14,8,22)],

Object Id, Shape Id, Origin, Start, Duration, End
Additional attributes (type, weight, customer, …) can eventually be added

sbox(1,[0,0],[2,1]), sbox(1,[0,1],[1,2]), sbox(1,[1,2],[3,1]),
sbox(2,[0,0],[3,1]), sbox(2,[0,1],[1,3]), sbox(2,[2,1],[1,1]),
sbox(3,[0,0],[2,1]), sbox(3,[1,1],[1,2]), sbox(3,[2,2],[3,1]),
sbox(4,[0,0],[3,1]), sbox(4,[0,1],[1,1]), sbox(4,[2,1],[1,3]),
sbox(5,[0,0],[2,1]), sbox(5,[1,1],[1,1]), sbox(5,[0,2],[2,1]),
sbox(6,[0,0],[3,1]), sbox(6,[0,1],[1,1]), sbox(6,[2,1],[1,1]),
sbox(7,[0,0],[3,2]),
sbox(8,[0,0],[2,3]),
sbox(9,[0,0],[1,4])]

Potential shapes, where a shape is defined by a set of sboxes sharing the same shape id

List of external constraints
(can eventually be added)

[non-overlapping([0,1],[1,2,3,4]), included([0,1],[1,2,3,4],[1,1],[5,4])]
Applications

(A) disjunctive

(B) machine assignment (e.g., parking assignment)

(C) machine assignment (machine dependant duration)

(D) 2D non-overlapping (fixed orientation)

(E) 2D non-overlapping (90° rotation)

(F) 2D non-overlapping (irregular shapes)

2D non-overlapping and assignment

3D non-overlapping

3D non-overlapping and assignment

pick-up delivery (time dimension)

gext(2, [object(1,1,[1,4]),1,3,4), object(2,2,[2,2]),1,2,3), object(3,1,[1,1]),1,1,2), object(4,3,[1,1],2,2,4), object(5,1,[2,3]),3,1,4), [sbox(1,[0,0],[2,1]), sbox(2,[0,0],[2,2]), sbox(3,[0,0],[1,3]), [non-overlapping([0,1],[1,2,3,4,5])])
EXAMPLE OF PROBLEM

Input: A set of parallelepipeds $P$ and a subset $P'$ of $P$

Constraints: (1) all parallelepipeds of $P$ should not overlap
(2) no parallelepipeds of $P'$ should be piled

Solution with geost: $\text{non-overlapping}([0, 1, 2], P)$
$\text{non-overlapping}([0, 1], P')$
Overall Architecture

EXTERNAL LAYER

EXTERNAL GEOMETRICAL CONSTRAINTS
- compatible (attributes, objects, pairs)
- included (attributes, objects, t, l)
- non-overlapping (attributes, objects)
- visible (attributes, objects, from, c)
  - InitFrameExternalConstraint (ctr, O, S)
  - GenInternalCts (ctr, o, O, S, FRAME)

INTERNAL LAYER

INTERNAL GEOMETRICAL CONSTRAINTS
- inbox (l, l)
- outbox (l, l)
- avoid_holes (n, nfr, slack, objects, first, last, noise, tabsize, fabvol, um)
- LexFeasible (ctr, minex, d, c, o): (bool, p)
- IsFeasible (ctr, min, d, k, o, c): (bool, f)
- CardInfeasible (ctr, k, o, c): (int)

KERNEL LAYER

LEXICOGRAPHIC SWEEP-POINT ALGORITHM

MAIN FUNCTIONS
- FilterCtrs (k, O, S, C): bool
- FilterObj (h, o, R, S): bool
- FruneMin (a, d, k, CTRS): bool
- FruneMax (a, d, k, CTRS): bool
- AdjustDown (s, n, o, d, k, f): (f, b, bool)
- AdjustUp (c, n, o, d, k, f): (f, b, bool)
- GetFR (d, k, o, C, increase): (bool, region)

EuroControl déc. 2008
Introduction

**External Geometrical Constraints**

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

*geost* on the web
Example of External Constraint: *distance*

Depending of the norm we consider we have different distances:

\[
\ell_q \text{ norm, } q \geq 1 \quad \|x\|_q = \left( \sum_{i=0}^{k-1} |x[i]|^q \right)^{\frac{1}{q}}
\]

\[
\ell_\infty \text{ norm} \quad \|x\|_\infty = \max_{0 \leq i < k} |x[i]|
\]

*More general distances also consider the assignment dimension*

*q=1: Manhattan
q=2: Euclidean

Handle minimum and maximum distances between objects of *geost* (manipulate only integers !)
Example of External Constraint: visible

IDEA

Given a set of potential observations places \( P \), and given for each box a set of visible faces, the **visible** constraint specifies that at least one visible face of each box should be entirely visible from at least one observation place of \( P \) at the start and end(-1) time associated to the box.

Completely visible faces from a set of observations points
Application of **visible**: pick-up delivery

time-dimension again!
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

geost on the web
Intermediate Layer

SERVICES ASSOCIATED TO AN INTERNAL CONSTRAINT (i.e., a set of forbidden points)

LexInfeasible \( (ictr, minlex, d, k, o) : (bool, p) \)

IsInfeasible \( (ictr, min, d, k, o, c) : (bool, P) \)

CardInfeasible \( (ictr, k, o) : (int) \)
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

**The Propagation Kernel**

A First Evaluation

Time (and trajectories)

*geost* on the web
Communication between Constraints

SLOGAN OF CONSTRAINT PROGRAMMING
Constraints communicate only via the domains of their shared variables.

ANOTHER APPROACH
A constraint can be assimilated as a set of forbidden points, each variable corresponding to a dimension

PROBLEM: hard to aggregate sets of forbidden points associated to different constraints !!!

SOLUTION: set of forbidden points associated to different constraint should communicate everything is handled in an implicit way (lazzy evaluation)
Sweep Algorithms in Computational Geometry

**Standard** technique for coming up with **efficient** algorithms

- Computational geometry, an introduction
  
  [Preparata & Shamos, 1985]

- Computational Geometry, Algorithms and Applications
  
  [Berg, Kreveld, Overmars & Schwarzkopf, 1997]

- Géométrie algorithmique
  
  [Boissonnat & Yvinec, 1995]
**Basic Idea of the Sweep Algorithm**
(in dimension 2)

Accumulates forbidden regions

\[
\begin{align*}
CTR_1(X, Y, \ldots) \\
CTR_2(X, Y, \ldots) \\
\vdots \\
CTR_n(X, Y, \ldots)
\end{align*}
\]

sharing 2 given variables X and Y

Is \(\min(X)\) feasible?
No, then move the sweep line.
Question: How to Generalize to $k$ Dimensions?

Key problem with the sweep-line status:

  don't want to use a multi-dimensional data structure since it just kills scalability
Geometric Kernel: a Lexicographic Sweep-Point Algorithm

VARIABLES
x1 in 1..4, y1 in 2..4
x2 in 4..4, y2 in 6..6
x3 in 2..4, y3 in 8..9
x4 in 7..7, y4 in 1..1
x5 in 1..8, y5 in 1..8, y5<>7

EXTERNAL CONSTRAINT (non-overlapping)
geoSt([object(1,1,[x1,y1],0,1,1), object(2,2,[x2,y2],0,1,1),
       object(3,3,[x3,y3],0,1,1), object(4,4,[x4,y4],0,1,1),
       object(5,5,[x5,y5],0,1,1)],
       [shape([0,2],[0,1]), shape([0,3],[0,1]), shape([0,1],[0,1]),
       shape([0,1],[0,3]), shape([0,5],[0,4])],
       [non-overlapping([0,1],[1,2,3,4,5])] )

INTERNAL CONSTRAINTS GENERATED FOR FILTERING THE ORIGIN OF THE FIFTH OBJECT, i.e. (x5,y5) (ICTRS)
ctr1: outbox([1,1],[2,2]) ctr3: outbox([1,8],[2,1])
ctr2: outbox([1,3],[6,4]) ctr4: outbox([3,1],[5,3])
ctr5: outbox([1,7],[8,1])

DCTRS (delayed internal constraint)
ctr1
ctr2
ctr3
ctr4
ctr5

ACTRS (active internal constraint)
ctr1
ctr2
ctr3
ctr4
ctr5

Sweep Point: c=(1,1)

DCTRS
ctr2
ctr5
ctr3
ctr4

ACTRS
ctr1

CONFLICT
ctr1

Sweep Point: c=(1,3)

DCTRS
ctr5
ctr3
ctr4

ACTRS
ctr1

CONFLICT
ctr2

(A)

(B)

(C)

(D)

(E)

(F)

EuroControl déc. 2008
Geometric Kernel: a Lexicographic Sweep-Point Algorithm

SWEEP POINT: c=(1,7)

DCTRS
- ctr3
- ctr4

ACTRS
- ctr5
- ctr2
- ctr1

CONFLICT
- ctr5

SWEEP POINT: c=(1,8)

DCTRS
- ctr3
- ctr5

ACTRS
- ctr3
- ctr5
- ctr2
- ctr1

SWEEP POINT: c=(3,1)

DCTRS

ACTRS
- ctr3
- ctr5
- ctr2
- ctr1

CONFLICT
- ctr3

SWEEP POINT: c=(3,4)

DCTRS

ACTRS
- ctr4
- ctr5
- ctr2
- ctr1

CONFLICT
- ctr2

SWEEP POINT: c=(3,7)

DCTRS

ACTRS
- ctr4
- ctr5
- ctr2
- ctr1

CONFLICT
- ctr5

SWEEP POINT: c=(3,8)

DCTRS

ACTRS
- ctr4
- ctr5
- ctr2
- ctr1

CONFLICT
- ctr2

EuroControl déc. 2008
Extensions Around the Kernel

GEOMETRICAL SWEEP KERNEL

Distance
- minimum distance
- maximum distance
- Manhattan, Euclidean

Non-overlapping
- vis-à-vis
- task intervals
- parity

Cumulative
- longest hole (bound)
- longest hole (exact)
- parallel conflict
- initial pruning
- two instants
- balancing knapsack

Disjunctive
- complete (small)
- generation (clique)

Symmetry
- lex chain (sweep)
- bounds (lex+non-overlapping)
- cumulative (compulsory part)
- cumulative (task interval)

Heuristics
- domination
- greedy mode
- placement heuristics
Where Splitting Objects Kills Propagation

(A) Shape to place within (B) so that it does not overlap rectangles r1, r2, r3

(B) Placement space where to put s

(C) Decomposing s in two rectangles r4 and r5

(D) Constraints of the problem

ctr1: x1+2=x2
ctr2: y1+1=y2
ctr3: r1 and r4 do not overlap
ctr4: r2 and r4 do not overlap
ctr5: r3 and r4 do not overlap
ctr6: r1 and r5 do not overlap
ctr7: r2 and r5 do not overlap
ctr8: r3 and r5 do not overlap
Where Splitting Objects Kills Propagation

Forbidden pairs of values for \((x_1, y_1)\) according to \(\text{ctr3}\)

Forbidden pairs of values for \((x_1, y_1)\) according to \(\text{ctr4}\)

Forbidden pairs of values for \((x_1, y_1)\) according to \(\text{ctr5}\)

Forbidden pairs of values for \((x_1, y_1)\) according to \(\text{ctr3}, \text{ctr4}\) and \(\text{ctr5}\)

Forbidden pairs of values for \((x_2, y_2)\) according to \(\text{ctr6}\)

Forbidden pairs of values for \((x_2, y_2)\) according to \(\text{ctr7}\)

Forbidden pairs of values for \((x_2, y_2)\) according to \(\text{ctr8}\)

Forbidden pairs of values for \((x_2, y_2)\) according to \(\text{ctr6}, \text{ctr7}\) and \(\text{ctr8}\)
**Where Splitting Objects Kills Propagation**

**QUESTION:**
How to combine information from \((x_1,y_1)\) and \((x_2,y_2)\) ?

**ANSWER:**
Combining the infeasible points for \((x_1,y_1)\) and \((x_2,y_2)\) ONLY possible if \(ctr_1\) and \(ctr_2\) are integrated within the sweep process!

\(ctr_1: x_1 + 2 = x_2\)
\(ctr_2: y_1 + 1 = y_2\)

Transmission of the forbidden pairs of values for \((x_2, y_2)\) to forbidden pairs of values for \((x_1, y_1)\) through the external constraints \(ctr_1\) and \(ctr_2\) is not done if \(ctr_1\) and \(ctr_2\) are not integrated within the sweep algorithm.

Overall set of forbidden pair of values for \((x_1, y_1)\)
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

*geost* on the web
A First Evaluation
with a focus on non-overlapping

- **Scalability** on loosely constrained problems (20% spare space)

- **Tight** placement problems (0% spare space)
  - Perfect squared squares
  - 3D pentominoes [Colmerauer, Gillet 99]

- **State of the art OR** for 2D orthogonal packing [Clautiaux, Carlier, Jouglet 07]
Loosely Constrained Problems

- Search first solution for random problem instances of $m \times k$-dimensional boxes for $k$ in {2,3,4} involving $t$ in {1,16,256,1024} distinct types of boxes, and $m$ in {1024,2048,…,65536}.

- The number of $1,048,576$ variables in geost was reached (first time in a constraint solver in a backtracking environment!).

- Can typically pack 1024 2D, 3D and 4D distinct boxes in at most 200 msec.

- Worst time, 13694 sec, obtained for packing 262.144 4D parallelepipeds corresponding to 1024 distinct types, with a memory consumption of 351MB.

- Approach sensible to the number of distinct types of boxes.

Results for $k=2$ for various $t$ and $m$ (time in msec)

ON THE WAY: MEMOIZATION SHOULD ENHANCE THESE RESULTS WHEN NOT TOO MANY DISTINCT SIZES!
Tight Placement Problems

Squared squares

Smallest square

Smallest surface

3-D pentomino (Giletta)

Partridge

Loading/unloading and visibility

Shikaku
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

geost on the web
Handling time

Motivation: a lot of practical problems mix geometrical constraints with time:
(1) Constructing a 3D packing,
(2) Pick-up delivery (objects stays between two dates),
(3) Containers management in a harbour (containers are organised in piles and stay between two dates),
(4) Planning trajectory (several consecutive moves for one object).

Extending geost

• **Solution**: add 3 (or 2) extra dimensions \((origin, duration, end)\) to the sweep algorithm.
• **Warning**: when the duration is variable the origin and end attributes are not equivalent!
• **Warning**: special care if one wants to generate the exact forbidden region with respect to the constraint \(origin + duration = end\) (using boxes kills the propagation since too many boxes).
• Trajectory made up from several objects: handle also **time** and **space continuity** constraints as well as **velocity**.
Introduction

External Geometrical Constraints

Internal Geometrical Constraints

The Propagation Kernel

A First Evaluation

Time (and trajectories)

geost on the web
geost on the web

• Integrated by **EMN** and **SICS** within **CHOCO** and **SICStus**
• since Nov. 2008 within the **Global Constraints Catalog**
see [http://www.emn.fr/x-info/sdemasse/gccat/](http://www.emn.fr/x-info/sdemasse/gccat/)

(1) description of **geost** (*and other geometric constraints*)

(2) overview (*and pointers*) to **propagation** techniques

(3) overview and illustration of placement **heuristics**

(4) overview of placement **problems**

(5) **xml** format for **geost** and problem instances
Global Constraint Catalog

Corresponding author: Nicolas Beldiceanu nicolas.beldiceanu@emn.fr
Online version: Sophie Demasse sophie.demasse@emn.fr

Search by:

<table>
<thead>
<tr>
<th>NAME</th>
<th>Keyword</th>
<th>Meta-keyword</th>
<th>Argument pattern</th>
<th>Graph description</th>
<th>Bibliography</th>
<th>Index</th>
</tr>
</thead>
</table>

Keywords (ex: Assignment, Bound consistency, Soft constraint, ...) can be searched by Meta-keywords (ex: Application area, Filtering, Constraint type, ...)

About the catalogue

The catalogue presents a list of 313 global constraints issued from the literature in constraint programming and from popular constraint systems. The semantic of each constraint is given together with a description in terms of graph properties and/or automata.

The catalogue is periodically updated by Nicolas Beldiceanu, Mats Carlsson and Jean-Xavier Rampon. Feel free to contact the first author for any questions about the content of the catalogue.

Download the Global Constraint Catalogue in pdf format:
- the last working version (2008-11-15) (about 12 Mo)
- the edited version (2005-08) (Sicstus technical report, about 7 Mo)

About this website and the electronic catalogue

This website provides the online version of the catalogue. As the pdf version, it is generated from the Prolog and LaTeX source files of the document. The online version is first conceived to ease the search through the catalogue: constraints can be searched by name, keyword, author, graph description, etc. Moreover, it
Changelog

2008-11-15 working version update: 313 constraints
- new constraints: all_equal and soft_all_equal_ctr, hard and soft equality constraints
- electronic catalogue: all used models of puzzle problem instances are provided in XML and Prolog format packing almost squares, Partridge, pentomino, smallest square for packing consecutive dominoes, smallest rectangle for packing rectangles, smallest square for packing rectangles.
- graphical view of the resolution of squared squares problems using of the geost constraint.
- permanent url adress of each keyword page: just add a "K" prefix and ".html" suffix to the keyword name (see e.g. Kcompulsory_part.html)

2008-09-18 working version update: 311 constraints
- new constraints: geost, a generic geometrical constraint for a large variety of puzzles, packing and placement problems; in_intervals, alldifferent_consecutive_values
- electronic catalogue: the example instance of each constraint is provided as an XML file, available on the constraint page. Helmut Simonis conceived the XML schema of the catalog.
- permanent url of each constraint page: just add a "C" prefix and ".html" suffix to the constraint name (see e.g. Calldifferent.html)

2008-02-03 working version update: 308 constraints
- new constraints: geometric constraints, arithmetic constraints,...
- electronic catalogue: the prolog description files (.pl) are now available on each constraint page, as well as the printable version of the page [.pdf] and the graphical specification of the constraint [.png]
- index: the general index of the catalogue with back references

2006-09-30 working version update: 276 constraints
- new constraints: open constraints
- electronic catalogue: prolog source files available
- biblio: index of the bibliographic citations with back references
- new arrangement for the constraint descriptions
- scaled delimiters for the multi-line formulæ

2006-06-12 working version online: 270 constraints
- introduction page
- figures automatically generated and resized
- page names = section numbers

2006-05-12
- constraints indexed on the elements (characteristics, restrictions, arc/set generators) of their graph description
- fix broken internal links / numbering / anchors

geost on the web
### 4.122. geost

**DESCRIPTION**

- **Origin**
  - Generalisation of `diffn`.

- **Constraint**
  - `geost(K, OBJECTS, SBOXES)`

**Type(s)**

- `VARIABLES` `collection(y-dvar)`
- `INTEGERS` `collection(y-int)`
- `POSITIVES` `collection(y-int)`

**Argument(s)**

- `K` `int`
- `OBJECTS` `collection(oid-int, sid-dvar, x-VARIABLES)`
- `SBOXES` `collection(sid-int, t-INTEGERS, l-POSITIVES)`

**Restriction(s)**

- `required(VARIABLES, v)`
- `required(INTEGERS, v)`
- `required(POSITIVES, v)`
- `required(OBJECTS, [oid, sid, x])`
- `K <= 0`
Conclusion

Once again, use the sweep idea: quite simple, but powerful!

- The overall architecture was designed in order to allow to integrate additional constraints without modifying the kernel.
- Can directly handle objects that move in time.
- When propagating on one object, consider all constraints involving that object.
- Scale better (one million integer variables in a standard constraint system: compatible with backtracking).

One last observation:

disjunctive, cumulative, non-overlapping constraints should all be integrated within one single global constraint since:

(1) they all correspond to related nested dynamic sub problems,
(2) allow to get better propagation,
(3) allow to reuse code.

EuroControl déc. 2008


JOURNAL VERSION SUBMITTED


INVITED FOR A SPECIAL ISSUE OF THE Constraint JOURNAL