Experimenting with Robotic Intra-Logistics Domains

Martin Gebser ¹  Philipp Obermeier ¹  Thomas Otto ¹  Torsten Schaub ¹  Orkunt Sabuncu ²  Van Nyugen ³  Tran Cao Son ³

¹University of Potsdam, Germany
²TED University, Ankara, Turkey
³New Mexico State University, Las Cruces, USA

Potassco
Introduction

1. Introduction
   - Motivation
   - Robotic Intra-Logistics

2. Benchmark Suite
   - Overview
   - Domains

3. Exemplary Evaluation
   - Instances
   - Encodings
   - Results

4. Outlook
Answer Set Programming
Answer Set Programming

- Declarative problem solving for combinatorial problems
Answer Set Programming

- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry
Answer Set Programming

- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry
Answer Set Programming

- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps

Logistics, manufacturing, automation, scheduling, etc.

Large instance sizes

Complex processes
Answer Set Programming
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps
- Logistics, manufacturing, automation, scheduling, etc.
Introduction

Motivation

Answer Set Programming
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps
- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes

Gebser et al. (KRR@UP) Experimenting with Robotic Intra-Logistics
Motivation

Answer Set Programming
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps
- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes
- Complex processes
Introduction

Answer Set Programming
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Motivation

Dynamic Real-World Apps
- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes
- Complex processes
Motivation

Answer Set Programming
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps
- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes
- Complex processes
Introduction

Motivation

Answer Set Programming

- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

Dynamic Real-World Apps

- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes
- Complex processes

⚡ Lack of Real-Life Test Data ⚡

- Existing benchmark suites kept intentionally simplistic
- No industrial scale test data in the public domain
**Answer Set Programming**
- Declarative problem solving for combinatorial problems
- Large spectrum of applications in academia and industry

**Dynamic Real-World Apps**
- Logistics, manufacturing, automation, scheduling, etc.
- Large instance sizes
- Complex processes

**Robotic Intra-Logistics as Benchmark Domain**
- Hard, dynamic planning problem related to MAS, scheduling, temporal logics, CSP, uncertainty, etc.
- Key concern of industry 4.0
Robotic Intra-Logistics

Robotic systems for logistics and warehouse automation based on hundreds of
- mobile robots
- movable shelves
Robotic Intra-Logistics

- Robotics systems for logistics and warehouse automation based on hundreds of mobile robots and movable shelves.
- Main tasks: order fulfillment, i.e.
  - routing
  - order picking
  - replenishment
Robotic Intra-Logistics

- Robotics systems for logistics and warehouse automation based on hundreds of
  - mobile robots
  - movable shelves
- Main tasks: order fulfillment, i.e.
  - routing
  - order picking
  - replenishment
- Many competing industry solutions:
  - Amazon, Dematic, Genzebach, Gray Orange, Swisslog
Robotic Intra-Logistics
Outline I

1 Introduction
   • Motivation
   • Robotic Intra-Logistics

2 Benchmark Suite
   • Overview
   • Domains

3 Exemplary Evaluation
   • Instances
   • Encodings
   • Results

4 Outlook
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
  - Concise problem specification
  - Domains ranging from MAPF to full order fulfillment

Resources

Website at http://potassco.org/asprilo
ICLP’18 paper, also available at https://arxiv.org/abs/1804.10247
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
asprilo Benchmark Suite

Main Components

- **Standardized benchmark domains**
- **Versatile instance generator**
  - Rich set of customization options
  - Leverages multi-shot ASP for generation
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
  - Animated playback of plans
  - Graphical editor for instances

Resources

Website at http://potassco.org/asprilo
ICLP'18 paper, also available at https://arxiv.org/abs/1804.10247
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
- Solution checker with error feedback

Resources

Website at http://potassco.org/asprilo
ICLP'18 paper, also available at https://arxiv.org/abs/1804.10247
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
- Solution checker with error feedback
  - Specific error descriptions
  - Modular design, easily extensible

Resources

Website at http://potassco.org/asprilo
ICLP'18 paper, also available at https://arxiv.org/abs/1804.10247
**asprilo Benchmark Suite**

**Main Components**

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
- Solution checker with error feedback
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
- Solution checker with error feedback

Resources

- Website at http://potassco.org/asprilo
asprilo Benchmark Suite

Main Components

- Standardized benchmark domains
- Versatile instance generator
- Visualizer for instances and plans
- Solution checker with error feedback

Resources

- Website at http://potassco.org/asprilo
- ICLP’18 paper, also available at https://arxiv.org/abs/1804.10247
General Domain A

- The warehouse is laid out as a (partial) 2-dimensional grid
- Shelves store products in a certain quantity, each shelf occupies a single grid node
- Mobile robots move and navigate through the warehouse along the grid, can carry shelves and deliver product units to picking stations
General Domain A

- **Highway nodes** are special grid nodes where robots must never put down a shelf.
- A set of **orders** is initially provided, an order is fulfilled if all its requested product units are delivered to its assigned picking station.
- **Main Goal:** plan robot actions such that all orders will be fulfilled.
Domain A Demo
Domains A, B, C, M

Domain A **most general domain**
Domains A, B, C, M

Domain A  most general domain

Domain B  ignores product quantities
Domains A, B, C, M

Domain A  most general domain

Domain B  ignores product quantities

Domain C  ignores product quantities
delivery actions at once
Domains A, B, C, M

Domain A  most general domain

Domain B  ignores product quantities

Domain C  ignores product quantities
            delivery actions at once

Domain M  only move actions
            singleton orders and shelves
            reach shelves with ordered products
Domain M Demo
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*

Test instances created with the generator 
Referential encodings for *asprilo*'s domains
Detailed setup description, instances, encodings and results available 
at [http://potassco.org/asprilo/experiments](http://potassco.org/asprilo/experiments)
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
  2. What is the impact of increasingly complex domains?

Test instances created with the generator

Referential encodings for *asprilo*'s domains

Detailed setup description, instances, encodings and results available at http://potassco.org/asprilo/experiments
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
  2. What is the impact of increasingly complex domains?
  3. What is the impact of decoupling sources of combinatorics?
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
  2. What is the impact of increasingly complex domains?
  3. What is the impact of decoupling sources of combinatorics?
- Test instances created with the generator

Test instances created with the generator.
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
  2. What is the impact of increasingly complex domains?
  3. What is the impact of decoupling sources of combinatorics?
- Test instances created with the generator
- Referential encodings for *asprilo*’s domains

Referential encodings for *asprilo*’s domains available at [http://potassco.org/asprilo/experiments](http://potassco.org/asprilo/experiments)
Exemplary Evaluation

- Exemplary benchmark evaluation to showcase *asprilo*
- Key questions of the analysis
  1. What is the impact of different representations of grid positions?
  2. What is the impact of increasingly complex domains?
  3. What is the impact of decoupling sources of combinatorics?
- Test instances created with the generator
- Referential encodings for *asprilo*’s domains
- Detailed setup description, instances, encodings and results available at [http://potassco.org/asprilo/experiments](http://potassco.org/asprilo/experiments)
## Instances

<table>
<thead>
<tr>
<th>Name</th>
<th>Generator Call and Resulting Layout</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>small:</strong></td>
<td>(\text{gen -x 11 -y 6 -X 4 -Y 2 -p 1 -s 16 -P 16 -u 16 -H --prs 1 -r 2 -o 2})</td>
</tr>
<tr>
<td><img src="image" alt="Small Layout" /></td>
<td></td>
</tr>
<tr>
<td><strong>medium:</strong></td>
<td>(\text{gen -x 19 -y 9 -X 5 -Y 2 -p 3 -s 60 -P 60 -u 60 -H --prs 1 -r 5 -o 5})</td>
</tr>
<tr>
<td><img src="image" alt="Medium Layout" /></td>
<td></td>
</tr>
<tr>
<td><strong>large:</strong></td>
<td>(\text{gen -x 46 -y 15 -X 8 -Y 2 -p 10 -s 320 -P 320 -u 320 -H --prs 1 -r 12 -o 12})</td>
</tr>
<tr>
<td><img src="image" alt="Large Layout" /></td>
<td></td>
</tr>
</tbody>
</table>
clingo Encoding for Domain M

routing

time(1..horizon).

direction((X,Y)) :- X=-1..1, Y=-1..1, |X+Y|=1.
nextto((X,Y),(X’,Y’),(X+X’,Y+Y’)) :- position((X,Y)), direction((X’,Y’)), position((X+X’,Y+Y’)).

\{ move(R,D,T) : direction(D) \} 1 :- isRobot(R), time(T).

     :- move(R,D,T), position(R,C,T-1), not nextto(C,D,_).

position(R,C,T) :- position(R,C,T-1), not move(R,_,T), isRobot(R), time(T).  \%inertia

moveto(C’,C,T) :- nextto(C’,D,C), position(R,C’,T-1), move(R,D,T).  \% edge collision
     :- moveto(C’,C,T), moveto(C,C’,T), C < C’.

\:- \{ position(R,C,T) : isRobot(R) \} > 1, position(C), time(T).  \% vertex collision
clingo Encoding for Domain M  
routing to shelves

time(1..horizon).

direction((X,Y)) :- X=-1..1, Y=-1..1, |X+Y|=1.
nextto((X,Y),(X’,Y’),(X+X’,Y+Y’)) :- position((X,Y)), direction((X’,Y’)), position((X+X’,Y+Y’)).

{ move(R,D,T) : direction(D) } 1 :- isRobot(R), time(T).

                      :- move(R,D,T), position(R, C’, T-1), not nextto(C’,D,_).

position(R,C,T) :- position(R,C,T-1), not move(R,_,T), isRobot(R), time(T). % inertia


:- { position(R,C,T) : isRobot(R) } > 1, position(C), time(T). % vertex collision

processed(O,A) :- ordered(O,A), shelved(S,A), position(S,C,0), position(R,C,horizon), isRobot(R).

processed(O) :- isOrder(O), processed(O,A) : ordered(O,A).

:- not processed(O), isOrder(O).
clingo Encoding for Domain A

routing + transport + delivery

time(1..horizon).

direction([(X,Y)] :- X=-1..1, Y=-1..1, |X+Y|=1.
nextto((X,Y),(X',Y'),(X+X',Y+Y')) :- position((X,Y)), direction((X',Y')), position((X+X',Y+Y')).

{ move(R,D,T) : direction(D) ;
  pickup(R,S,T) : isShelf(S) ;
  putdown(R,S,T) : isShelf(S) } 1 :- isRobot(R), time(T).

waits(R,T) :- not pickup(R,_,T), not putdown(R,_,T), not move(R,_,T), isRobot(R), time(T).

position(R,C,T) :- move(R,D,T), position(R,C',T-1), nextto(C',D,C).
  :- move(R,D,T), position(R,C ,T-1), not nextto(C, D, _).

  :- pickup(R,S,T), carries(R,_,T-1).
  :- pickup(R,S,T), carries(_,S,T-1).
  :- pickup(R,S,T), position(R,C,T-1), position(S,C',T-1), C != C'.

  :- putdown(R,S,T), not carries(R,S,T-1).

serves(R,S,P,T) :- position(R,C,T), carries(R,S,T), position(P,C), isStation(P).

position(R,C,T) :- position(R,C,T-1), not move(R,_,T), isRobot(R), time(T).

position(S,C,T) :- position(R,C,T-1), not putdown(R,_,T), time(T).

position(S,C,T) :- position(S,C,T-1), not carries(_,S,T), isShelf(S), time(T).

moveto(C',C,T) :- nextto(C',D,C), position(R,C',T-1), move(R,D,T).
  :- moveto(C',C,T), moveto(C,C',T), C < C'.

  :- { position(R,C,T) : isRobot(R) } > 1, position(C), time(T).
  :- { position(S,C,T) : isShelf(S) } > 1, position(C), time(T).
## Encoding Variants

### Variants

<table>
<thead>
<tr>
<th>Encodings</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>clingo</code></td>
<td>boolean encoding</td>
</tr>
<tr>
<td><code>clingo_{xy}</code></td>
<td>boolean encoding + split positional coordinates</td>
</tr>
<tr>
<td><code>clingcon</code></td>
<td>linear constraints for positions and product quantities</td>
</tr>
<tr>
<td><code>clingo[DL]</code></td>
<td>difference constraints for positions and product quantities</td>
</tr>
</tbody>
</table>

### Task Assignment

- Robots assigned a subset of shelves and picking stations
- All variants were tested with and without task assignments.
Experimental results in average run time & number of timeouts

<table>
<thead>
<tr>
<th>domain</th>
<th>makespan</th>
<th>encoding</th>
<th>small</th>
<th>medium</th>
<th>large</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>6/10/25</td>
<td>clingo</td>
<td>0(0)</td>
<td>0(0)</td>
<td>73(4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;xy&lt;/sub&gt;</td>
<td>0(0)</td>
<td>16(1)</td>
<td>591(14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>0(0)</td>
<td>37(0)</td>
<td>1168(52)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;[DL]&lt;/sub&gt;</td>
<td>0(0)</td>
<td>193(1)</td>
<td>1648(96)</td>
</tr>
<tr>
<td>M&lt;sub&gt;a&lt;/sub&gt;</td>
<td>6/10/25</td>
<td>clingo</td>
<td>0(0)</td>
<td>0(0)</td>
<td>41(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;xy&lt;/sub&gt;</td>
<td>0(0)</td>
<td>0(0)</td>
<td>763(27)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>0(0)</td>
<td>36(0)</td>
<td>1163(49)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;[DL]&lt;/sub&gt;</td>
<td>0(0)</td>
<td>86(1)</td>
<td>1679(102)</td>
</tr>
<tr>
<td>C&lt;sup&gt;M&lt;/sup&gt;</td>
<td>20/-/-</td>
<td>clingo</td>
<td>805(40)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>695(30)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td>C&lt;sup&gt;M&lt;/sup&gt;&lt;sub&gt;a&lt;/sub&gt;</td>
<td>21/35/-</td>
<td>clingo</td>
<td>23(1)</td>
<td>370(5)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>38(2)</td>
<td>459(15)</td>
<td>1800(120)</td>
</tr>
<tr>
<td>B&lt;sup&gt;M&lt;/sup&gt;</td>
<td>26/-/-</td>
<td>clingo</td>
<td>970(53)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>807(37)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td>B&lt;sup&gt;M&lt;/sup&gt;&lt;sub&gt;a&lt;/sub&gt;</td>
<td>26/39/-</td>
<td>clingo</td>
<td>12(0)</td>
<td>566(19)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>29(0)</td>
<td>623(25)</td>
<td>1800(120)</td>
</tr>
<tr>
<td>A&lt;sup&gt;M&lt;/sup&gt;</td>
<td>26/-/-</td>
<td>clingo</td>
<td>984(55)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>856(41)</td>
<td>1800(120)</td>
<td>1800(120)</td>
</tr>
<tr>
<td>A&lt;sup&gt;M&lt;/sup&gt;&lt;sub&gt;a&lt;/sub&gt;</td>
<td>26/39/-</td>
<td>clingo</td>
<td>12(0)</td>
<td>577(18)</td>
<td>1800(120)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clingo&lt;sub&gt;con&lt;/sub&gt;</td>
<td>49(1)</td>
<td>625(22)</td>
<td>1800(120)</td>
</tr>
</tbody>
</table>
Further extending *asprilo* based on user feedback

http://potassco.org/asprilo
Further extending *asprilo* based on user feedback

Explore ASP design patterns and techniques

- scalability
- temporal logic
- preference handling
- uncertainty
- online processing

http://potassco.org/asprilo