Mixed Multi-Unit Combinatorial Auctions for Supply Chain Automation

Andrea Giovannucci
Meritxell Vinyals
Jesus Cerquides
Ulle Endriss
Juan Antonio Rodríguez-Aguilar
Pedro Meseguer

Institut d’Investigació en Intel·ligència Artificial (IIIA-CSIC)
Outline

- Motivation
- Background (MMUCA)
- Limitations of WD solvers for MMUCA
- The Improved Solver
- Empirical evaluation
- Future work
Motivations

- The organisational structure of enterprises is changing
- Increment of outsourced activity
- From monolithic to collaborative structures that tend to reduce their size
Chinese Motorbike Industry

- Small firms meet in online places and coffee shops
- Each one is assigned the task it is best at
- A self-organising system of design and production
Background

- Business partners are moving from the roles of suppliers, manufacturers, and customers to the role of collaborators.
- In this environment, the choice of the best business partners is critical.
Goals

- Design a selection and coordination process among multiple partners so that:
  - it is easy to automate
  - it meets particular production requirements
  - it optimises production costs
Motivations and Goals

Example

Sale Forecast

200 Apple Pies

Motivations and Goals

- Butter
- Sugar
- Flour
- Apple
- Margarine

Make Dough

Make Filling

Dough

Filling

Baking

Apple Pie

€ 5

€ 6

€ 14
Procurement Stage

Motivations and Goals

- Procurement Stage
  - butter
  - sugar
  - flour
  - apple
  - margarine

  - make dough
  - make filling
  - baking

- Motivations and Goals

- €5
- €6
- €14
- Apple Pie
Make-or-Buy

Motivations and Goals
Procurement Stage

Motivations and Goals

- Butter
- Sugar
- Flour
- Apple
- Margarine

1. Make Dough
   - €5

2. Make Filling
   - €6

3. Baking
   - €14

Apple Pie
Motivations and Goals

Make-or-Buy-or-Collaborate

- Butter
- Sugar
- Flour
- Apple
- Margarine

Make Dough

Make Filling

Dough

Filling

Baking

€ 5

€ 6

€ 14

Apple Pie
Motivations and Goals

Make-or-Buy-or-Collaborate

- Mixed Multiunit Combinatorial Auctions (MMUCA)

- Automatically selects the best Make-or-Buy-or-Collaborate decisions
Overview

- Bidding Language (IJCAI 07)
- Winner Determination Problem
  (1) Definition (IJCAI 07)
  (2) Solvers
    • Petri-Nets based (AAMAS 07)
    • Direct Integer Programming (IJCAI 07)
    • Connected Component Integer Program (AAMAS 08)
- MMUCA
  (2) Solvers
- Empirical Evaluation (IJA 08)
Outline

- Motivation
- Background (MMUCA)
- Limitations of WDP solvers for MMUCA
- The Improved Solver
- Empirical evaluation
- Future work
Mixed Multi-unit Combinatorial Auctions

- An extension of Combinatorial Auctions that provides:
  - A formal language to express preferences over operations across the supply chain
  - A formalisation of the optimisation problem that selects:
    - (1) The best business partners
    - (2) A feasible sequence of operations

Automatically selects the best Make-or-Buy-or-Collaborate decisions
Atomic Bid and Supply Chain Operation

$SCO = (\text{Inputs, Outputs})$

1. $SCO_4 = (2'H_2O, 1'O_2 + 2'H_2)$
2. $SCO_5 = (1'O_2 + 2'H_2, \text{nothing})$

$BID_1 = (1'SCO_1 + 2'SCO_2, -€2)$

$BID_1 \text{ XOR } BID_2 \text{ XOR } BID_3 \text{ XOR } BID_4$

$BID_1 \text{ OR } BID_2 \text{ OR } BID_3 \text{ OR } BID_4$
A bidder can express preferences over bundles of SCOs (Atomic Bid).

A bidder can submit combinations of Atomic Bids (e.g. XOR, OR).

Theorem: XOR is expressive enough to represent any valuation.
Solution: $\langle \text{SCO}_1 \rangle$

Solution: $\langle \text{SCO}_1, \text{SCO}_3 \rangle$

Solution: $\langle \text{SCO}_1, \text{SCO}_3, \text{SCO}_6 \rangle$

Revenue:

$\text{€} -10 - \text{€} 8 + \text{€} 25 = +7$
Compute a sequence of SCOs selected among the ones submitted by bidders such that:

- it fulfils the constraints expressed by the bids
- it is feasible
- it maximises the auctioneer's revenue
Outline

- Motivation
- Background (MMUCA)
  - Limitations of WD solvers for MMUCA
- The Improved Solver
- Empirical evaluation
- Future work
Comparing solvers for MMUCA

<table>
<thead>
<tr>
<th>SOLVER</th>
<th>TOPOLOGY</th>
<th>#Decision Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Petri-Nets Based Integer Program</td>
<td>ACYCLIC</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Direct Integer Program</td>
<td>ANY</td>
<td>$O(N^2)$</td>
</tr>
<tr>
<td>Connected Components IP</td>
<td>ANY</td>
<td>$O(N) \leq \ll O(N^2)$</td>
</tr>
</tbody>
</table>

$N$: overall number of Supply Chain Operations
Cyclic topologies
Cyclic topologies

For instance

Resource reuse
Production Cycles
Direct Integer Program

\[
\begin{array}{cccc}
& SCO_1 & & \\
1 & & & \\
SCO_0 & & SCO_3 & SCO_2 \\
X2 & & & \\
& SCO_4 \\
\end{array}
\]

<table>
<thead>
<tr>
<th>Positions</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SCO_0</td>
</tr>
<tr>
<td>2</td>
<td>SCO_1</td>
</tr>
<tr>
<td>3</td>
<td>SCO_1</td>
</tr>
<tr>
<td>4</td>
<td>SCO_1</td>
</tr>
<tr>
<td>5</td>
<td>SCO_1</td>
</tr>
<tr>
<td>6</td>
<td>SCO_1</td>
</tr>
</tbody>
</table>
Direct Integer Programming approach

<table>
<thead>
<tr>
<th>Positions</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SCO₀</td>
</tr>
<tr>
<td>2</td>
<td>SCO₀</td>
</tr>
<tr>
<td>3</td>
<td>SCO₀</td>
</tr>
<tr>
<td>4</td>
<td>SCO₀</td>
</tr>
<tr>
<td>5</td>
<td>SCO₀</td>
</tr>
<tr>
<td>6</td>
<td>SCO₀</td>
</tr>
</tbody>
</table>
## DIP explained

<table>
<thead>
<tr>
<th>Positions</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCO</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SCO1</td>
<td>SCO1</td>
<td>SCO1</td>
<td>SCO1</td>
<td>SCO1</td>
<td>SCO1</td>
</tr>
<tr>
<td>2</td>
<td>SCO2</td>
<td>SCO2</td>
<td>SCO2</td>
<td>SCO2</td>
<td>SCO2</td>
<td>SCO2</td>
</tr>
<tr>
<td>3</td>
<td>SCO3</td>
<td>SCO3</td>
<td>SCO3</td>
<td>SCO3</td>
<td>SCO3</td>
<td>SCO3</td>
</tr>
<tr>
<td>4</td>
<td>SCO4</td>
<td>SCO4</td>
<td>SCO4</td>
<td>SCO4</td>
<td>SCO4</td>
<td>SCO4</td>
</tr>
<tr>
<td>5</td>
<td>SCO5</td>
<td>SCO5</td>
<td>SCO5</td>
<td>SCO5</td>
<td>SCO5</td>
<td>SCO5</td>
</tr>
<tr>
<td>6</td>
<td>SCO6</td>
<td>SCO6</td>
<td>SCO6</td>
<td>SCO6</td>
<td>SCO6</td>
<td>SCO6</td>
</tr>
</tbody>
</table>
Problem

- The search space associated to DIP is big
- This affects the computational performance of DIP
- Can we reduce the associated search space?
Outline

- Motivation
- Background (MMUCA)
- Limitations of WD solvers for MMUCA
- The Improved Solver
- Empirical evaluation
- Future work
Equivalent Solutions

Solution sequence:

SCO₀, SCO₁, SCO₂, SCO₃, SCO₄
Solution sequence:

SCO₁, SCO₂, SCO₀, SCO₂

SCO₁, SCO₀, SCO₂, SCO₂

SCO₀, SCO₁, SCO₂, SCO₂

SCO₁, SCO₂, SCO₀, SCO₂

SCO₀, SCO₁, SCO₂, SCO₂

The improved Solver - CCIP
Reducing the search space

• Can we avoid considering re-orderings of the solution sequence?

• Indeed: Assume that the auctioneer doesn’t care about the ordering of a solution sequence as long as enough goods are available for every SCO in the sequence.
Equivalent Sequences

Solution sequence:

- $SCO_1, SCO_2, SCO_0, SCO_2$
- $SCO_1, SCO_2, SCO_0, SCO_2$
- $SCO_0, SCO_1, SCO_2, SCO_2$
- $SCO_0, SCO_1, SCO_2, SCO_2$
How to remove some sequences

- Each solution to the MMUCA WDP can be reordered into a solution that complies with a given TEMPLATE.

- This template is built considering the dependency relationships among SCOs.
The improved Solver - CCIP

The dependency graph

SCO Dependency Graph

SCO\textsubscript{0} \rightarrow SCO\textsubscript{1} \rightarrow SCO\textsubscript{2} \rightarrow SCO\textsubscript{3} \rightarrow SCO\textsubscript{4}
SCO Dependency Graph

SCO\textsubscript{0} depends on SCO\textsubscript{0}, SCO\textsubscript{1}

SCO\textsubscript{2} depends on SCO\textsubscript{0}, SCO\textsubscript{1}

SCO\textsubscript{3}

SCO\textsubscript{4}

SCO\textsubscript{2}
SCO Dependency Graph

SCO
depends on 

SCO
depends on 

SCO , SCO , SCO , SCO , SCO

SCO , SCO , SCO , SCO , SCO

SCO , SCO , SCO , SCO , SCO
SCO Dependency Graph

SCO₀ and SCO₀ are independent

SCO₁
1
SCO₀

SCO₃

X₂
SCO₂

SCO₂

SCO₃
SCO₄

SCO₄

The improved Solver - CCIP
The dependency graph
SCO Dependency Graph

SCO₁ and SCO₀ are independent

SCO₁, SCO₂, SCO₀, SCO₂
SCO₀, SCO₂, SCO₁, SCO₂

The improved Solver - CCIP

The dependency graph
SCO Dependency Graph

- SCO₂ depends on SCO₄
- SCO₄ depends on SCO₂
- SCO₂, SCO₄ belong to a loop

The improved Solver - CCIP
Strongly Connected Components

\[ \text{SCO}_2, \text{SCO}_3, \text{SCO}_4 \]
cannot be ordered among them

We group them: \text{SCCs}
Strongly Connected Components

<table>
<thead>
<tr>
<th>Pos</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \text{SCO}_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( \text{SCO}_0 )</td>
</tr>
<tr>
<td>3</td>
<td>( \text{SCO}_2, \text{SCO}_3, \text{SCO}_4 )</td>
</tr>
<tr>
<td>4</td>
<td>( \text{SCO}_2, \text{SCO}_3, \text{SCO}_4 )</td>
</tr>
<tr>
<td>5</td>
<td>( \text{SCO}_2, \text{SCO}_3, \text{SCO}_4 )</td>
</tr>
<tr>
<td>6</td>
<td>( \text{SCO}_2, \text{SCO}_3, \text{SCO}_4 )</td>
</tr>
</tbody>
</table>

The dependency graph:
## Strongly Connected Components

<table>
<thead>
<tr>
<th>Pos</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$SCO_1$</td>
</tr>
<tr>
<td>2</td>
<td>$SCO_0$</td>
</tr>
<tr>
<td>3</td>
<td>$SCO_2,SCO_3,SCO_4$</td>
</tr>
<tr>
<td>4</td>
<td>$SCO_2,SCO_3,SCO_4$</td>
</tr>
<tr>
<td>5</td>
<td>$SCO_2,SCO_3,SCO_4$</td>
</tr>
<tr>
<td>6</td>
<td>$SCO_2,SCO_3,SCO_4$</td>
</tr>
</tbody>
</table>
### The Solution Template

<table>
<thead>
<tr>
<th>Positions</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template</td>
<td>SCO₀</td>
<td>SCO₁</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
</tr>
<tr>
<td>SEQ A</td>
<td>SCO₀</td>
<td>SCO₁</td>
<td>SCO₃</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
</tr>
<tr>
<td>SEQ B</td>
<td>SCO₂</td>
<td>SCO₁</td>
<td>SCO₃</td>
<td>SCO₀</td>
<td>SCO₂</td>
<td>SCO₂</td>
</tr>
</tbody>
</table>

![Diagram of the solution template](attachment:image.png)

The improved Solver - CCIP

The solution template
Proof of correctness

- **THEOREM**: “each solution to the MMUCA WDP can be reordered into an equivalent solution that fulfils the solution template”

- If we reduce the search space to the sequences fulfilling the solution template we do not lose any solutions
Comparing DIP and CCIP

- The hypothesis behind DIP is that a SCO can hold any position within the solution sequence.

\[ 5 \times 6 = 30 \]

<table>
<thead>
<tr>
<th>Positions</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template</td>
<td>SCO₀</td>
<td>SCO₀</td>
<td>SCO₀</td>
<td>SCO₀</td>
<td>SCO₀</td>
<td>SCO₀</td>
</tr>
<tr>
<td></td>
<td>SCO₁</td>
<td>SCO₁</td>
<td>SCO₁</td>
<td>SCO₁</td>
<td>SCO₁</td>
<td>SCO₁</td>
</tr>
<tr>
<td></td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
</tr>
<tr>
<td></td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
</tr>
<tr>
<td></td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
</tr>
</tbody>
</table>

The improved Solver - CCIP
Comparing DIP and CCIP

- The hypothesis behind CCIP is that a SCO can hold only the positions allowed by the template.

<table>
<thead>
<tr>
<th>Positions</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template</td>
<td>SCO₀</td>
<td>SCO₁</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
<td>SCO₂</td>
</tr>
<tr>
<td></td>
<td>SCO₂</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
<td>SCO₃</td>
</tr>
<tr>
<td></td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
<td>SCO₄</td>
</tr>
</tbody>
</table>
## Comparing solvers

<table>
<thead>
<tr>
<th>SOLVER</th>
<th>TOPOLOGY</th>
<th>#Decision Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Petri-Nets Based</td>
<td>ACYCLIC</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Direct Integer Program</td>
<td>ANY</td>
<td>$O(N^2)$</td>
</tr>
<tr>
<td>Connected Component Integer Program (CCIP)</td>
<td>ANY</td>
<td>$O(k^2 \text{ SCC})$</td>
</tr>
</tbody>
</table>

$N$: overall number of Supply Chain Operations
Outline

- Motivation
- Background (MMUCA)
- Limitations of WD solvers for MMUCA
- The Improved Solver
  - Empirical evaluation
- Future work
(a) Components of a car engine. (b) Supply chain for a car's engine.
MMUCA WDP Generator

Empirical Evaluation
The improved Solver - CCIP

Empirical Evaluation

![Graph showing time vs. number of transformations for different parameter values.](image)
Conclusions

• The scalability of an IP implementation of MMUCA is affected by the size of the largest connected components.

• When there is a "natural" flow in the supply chain, CCIP scales reasonably well wrt number of transformations and goods.
Outline

- Motivation
- Background (MMUCA)
- Limitations of WD solvers for MMUCA
- The Improved Solver
- Empirical evaluation
- Future work
Future Work

- Incorporate time
  - time to perform operations
  - time to finish before a deadline
- Incorporate uncertainty
  - bidders may fail
  - maximise the expected value
- Study connections to Planning