Bounded Decentralised Coordination over Multiple Objectives

F.M. Delle Fave, R. Stranders, A. Rogers, N.R. Jennings
{fmdf08r, rs2, acr, nrj}@ecs.soton.ac.uk
This talk is about B-MOMS: an algorithm to solve multi-objective DCOPs.

Multi-Objective DCOPS

1) Bounding phase
2) Max-Sum phase
3) Value Propagation phase

Bounded Multi-Objective Max-Sum

Empirical Evaluation

M = nbr of agents
DCOPs are a well known tool to represent multi agent coordination problems
DCOPs are a well known tool to represent multi agent coordination problems.
DCOPs are a well known tool to represent multi agent coordination problems

Agents share a common goal
DCOPs research has addressed problems involving resource constraints.

**Problem:** agents have limited resources

**Example:** computation, communication, and battery life

**Consequence:** Optimal solution is very hard to compute
DCOPs research has addressed safety critical problems

Problem: risk of loss of life or damage to property

Example: search and rescue missions

Consequence: some guarantees on the solution quality are needed
However real world problems are also multi-objective

**Problem:** multiple (conflicting) objectives exist

**Example:** In search and rescue, agents need to search, track, and maintain communications

**Consequence:** algorithms need to carefully satisfy these multiple (conflicting) objectives
Indeed, current research does not meet all these requirements.

Life Critical
- B-MS, k-optimality

Multi-Objective
- MOBE, MOBnB

Resource Constrained
- MGM, DSA, Max-Sum

We need a new algorithm: B-MOMS
We define MO-DCOPs: constraint functions become constraint vectors

**MO-DCOPs:**

\[ U_1(x_1, x_2) = [U_{11}, U_{12}] \]

\[ U_2(x_2, x_3) = [U_{21}, U_{22}] \]

Local constraint vectors
We define MO-DCOPs: the global function become a global vector

**MO-DCOPs:**

$U_1(x_1, x_2) = [U_{11}, U_{12}]$

$U_2(x_2, x_3) = [U_{21}, U_{22}]$

Global constraint vector

$U(x) = \text{argmax} \sum U_i(x_i)$

$= \text{argmax} \sum [U_{i1}, U_{i2}]$
MO-DCOPs can be directly encoded into a factor graph

\[ U_1 = [U_{11}, U_{12}] \]

\[ U_2 = [U_{21}, U_{22}] \]
MO-DCOPS have multiple optimal solutions which are non-comparable

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$U_1 = [U_{11}, U_{12}]$</th>
<th>$U_2 = [U_{21}, U_{22}]$</th>
<th>$U = U_1 + U_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2)</td>
<td>(2,0)</td>
<td>(3,2)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1)</td>
<td>(0,2)</td>
<td>(2,3)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0)</td>
<td>(4,3)</td>
<td>(4,3)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1)</td>
<td>(2,3)</td>
<td>(3,4)</td>
</tr>
</tbody>
</table>

$U_1 = [U_{11}, U_{12}]$ and $U_2 = [U_{21}, U_{22}]$ are non-comparable, and $U = U_1 + U_2$.
MO-DCOPS have multiple optimal solutions which are non-comparable

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(U_1 = [U_{11}, U_{12}])</th>
<th>(U_2 = [U_{21}, U_{22}])</th>
<th>(U = U_1 + U_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2)</td>
<td>(2,0)</td>
<td>(3,2)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1)</td>
<td>(0,2)</td>
<td>(2,3)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0)</td>
<td>(4,3)</td>
<td>(4,3)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1)</td>
<td>(2,3)</td>
<td><strong>(3,4)</strong> dominates</td>
</tr>
</tbody>
</table>
MO-DCOPS have multiple optimal solutions which are non-comparable

\[
U_1 = [U_{11}, U_{12}] \quad U_2 = [U_{21}, U_{22}] \quad U = U_1 + U_2
\]

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(U_1)</th>
<th>(U_2)</th>
<th>(U)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2)</td>
<td>(2,0)</td>
<td>(3,2)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1)</td>
<td>(0,2)</td>
<td>(2,3)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0)</td>
<td>(4,3)</td>
<td>(4,3)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1)</td>
<td>(2,3)</td>
<td>(3,4)</td>
</tr>
</tbody>
</table>

Non-dominated vectors
MO-DCOPS have multiple optimal solutions which are non-comparable

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$U_1 = [U_{11}, U_{12}]$</th>
<th>$U_2 = [U_{21}, U_{22}]$</th>
<th>$U = U_1 + U_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2)</td>
<td>(2,0)</td>
<td>(3,2)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1)</td>
<td>(0,2)</td>
<td>(2,3)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0)</td>
<td>(4,3)</td>
<td>(4,3)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1)</td>
<td>(2,3)</td>
<td>(3,4)</td>
</tr>
</tbody>
</table>

Pareto optimal solutions:

*it is not possible to increase the value of one objective without decreasing the value of another.*
This talk is about B-MOMS: an algorithm to solve multi-objective DCOPs

1) Bounding phase
2) Max-Sum phase
3) Value Propagation phase

Multi-Objective DCOPS

Bounded Multi-Objective Max-Sum

Empirical Evaluation

M = nbr of agents
We developed the B-MOMS algorithm to solve MO-DCOPs

Bounded Multi-Objective Max-Sum

- Extends the Bounded Max-Sum Algorithm

- Proceeds in 3 phases:
  1. Bounding phase
  2. Max-Sum phase
  3. Value Propagation phase
Phase 1: The Bounding Phase provides quality guarantees

- Prune the factor graph to a tree to guarantee convergence of the max-sum algorithm
- Remove edges with minimal impact on solution quality

\[ U_1 = [U_{11}, U_{12}] \]
\[ U_2 = [U_{21}, U_{22}] \]
Phase 1: The Bounding Phase provides quality guarantees

- Prune the factor graph to a tree to guarantee convergence of the max-sum algorithm
- Remove edges with minimal impact on solution quality

\[ U_1 = [U_{11}, U_{12}] \]
\[ U_2' = [U_{21}', U_{22}'] \]

(details in the paper)
Phase 2: The Max-Sum Phase solves the approximated problem

\[ \mathbf{U}_1 = [\mathbf{U}_{11}, \mathbf{U}_{12}] \]

- Optimally solves the approximated problem (proof in the paper)
Phase 2: The Max-Sum Phase solves the approximated problem

- Messages flow between function and variable nodes of the factor graph
  - From variable to function
    \[ Q_{n\rightarrow m}(x_n) = \sum_{m' \in M(n) \setminus m} R_{m'\rightarrow n}(x_n) \]
  - From function to variable
    \[ R_{m\rightarrow n}(x_n) = \max_{x_m \setminus n} \left( U_m(x_m) + \sum_{n' \in N(m) \setminus n} Q_{n'\rightarrow m}(x_{n'}) \right) \]
Phase 2: The Max-Sum Phase solves the approximated problem

- Messages flow between function and variable nodes of the factor graph
  - From variable to function
    \[ Q_{n \rightarrow m}(x_n) = \sum_{m' \in M(n) \setminus m} R_{m' \rightarrow n}(x_n) \]
  - From function to variable
    \[ R_{m \rightarrow n}(x_n) = \max_{x_m \setminus n} \left( U_m(x_m) + \sum_{n' \in N(m) \setminus n} Q_{n' \rightarrow m}(x_{n'}) \right) \]

However, $\text{max}$ and $+$ operators are generalised to consider multiple objectives
Phase 2: Max-Sum messages now contain multiple non-dominated vectors

Example: computing the message from $U_1$ to $x_1$:

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$U_1 = [U_{11}, U_{12}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2) + (0,1)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1) + (1,0)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0) + (0,1)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1) + (1,0)</td>
</tr>
</tbody>
</table>
Phase 2: Max-Sum messages now contain multiple non-dominated vectors

Example: computing the message from $U_1$ to $x_1$:

$U_1 = [U_{11}, U_{12}]$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$U_1 + Q_{2\rightarrow 1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(1,2) + (0,1)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(2,1) + (1,0)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(0,0) + (0,1)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(1,1) + (1,0)</td>
</tr>
</tbody>
</table>

Result:

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$U_1 \rightarrow x_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(1,3), (3,1)</td>
</tr>
<tr>
<td>1</td>
<td>(2,1)</td>
</tr>
</tbody>
</table>
At the end of phase 2, each agent recovers its corresponding PO assignments.

\[ U_1 = [U_{11}, U_{12}] \]

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>PO</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(0,0)</td>
</tr>
<tr>
<td>1</td>
<td>(2,0), (1,1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( x_2 )</th>
<th>PO</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(2,0)</td>
</tr>
<tr>
<td>1</td>
<td>(1,1)</td>
</tr>
</tbody>
</table>
Phase 3: Value Propagation selects one Pareto optimal assignment

$U_1 = [U_{11}, U_{12}]$

Two solutions: $(1, 0)$ and $(1, 1)$

$U(x_1 = 1, x_2 = 0) = (2, 0)$
$U(x_1 = 1, x_2 = 1) = (1, 1)$
Phase 3: Value Propagation selects a consistent Pareto optimal assignment

Either at random, or based on a logical condition.

Example:
maximise objective 1, subject to objective 2 > X
This talk is about B-MOMS: an algorithm to solve multi-objective DCOPs

1) Bounding phase
2) Max-Sum phase
3) Value Propagation phase
To test our algorithm we generalise the graph colouring problem into a tri-objective problem

Objective 1: canonical conflict function

- \( U = -1 \)
- \( U = 0 \)
To test our algorithm we generalise the graph colouring problem into a tri-objective problem

Objective 2 - chromatic ordering:
Agents with a higher index shall have a higher colour index

\[ \begin{align*}
\text{Blue} &= 3 \\
\text{Red} &= 2 \\
\text{Green} &= 1
\end{align*} \]

An integer value is assigned to each colour

\( i, j \) are the agents indices

\[ \begin{align*}
\text{If } j > i \text{ then } & \quad \begin{cases} 
\text{Green} < \text{Red} & U = -1 \\
\text{Green} > \text{Red} & U = 0
\end{cases}
\end{align*} \]
To test our algorithm we generalise the graph colouring problem into a tri-objective problem

Objective 3 - chromatic distance:
The norm of two agents' colour indices needs to be equal to 1

\[ i, j \text{ are the agents indices} \]

\[
\begin{align*}
\text{If } j > i \text{ then} & \quad \|i - j\| = 3 - 3 = 0 \quad U = -1 \\
& \quad \|i - j\| = 2 - 1 = 1 \quad U = 0
\end{align*}
\]
Our results: B-MOMS typically provides solutions within 50% of the optimal
Our results: B-MOMS is optimal on acyclic graphs (proof in paper)

\[ M = \text{# agents} \]

\[ 1 = \text{optimal} \]
Our results: On acyclic graphs, the runtime is, on average 0.5 sec per agent for up to 100 agents.
Our results: Even on fully constrained problems, the runtime is < 30 minutes for 100 agents.
In conclusion, B-MOMS is an efficient algorithm for solving DCOPs with multiple objectives.
In conclusion, B-MOMS is an efficient algorithm for solving DCOPs with multiple objectives.

For future work, we will investigate problems with uncertain objective functions.