AAAI-18 Tutorial on:

MULTI-AGENT DISTRIBUTED CONSTRAINED OPTIMIZATION

Ferdinando Fioretto
University of Michigan

William Yeoh
Washington University at St. Louis

Roie Zivan
Ben-Gurion University of the Negev
SCHEDULE

- 11:20am: Preliminaries
- 11:40am: DCOP Algorithms
- 12:20pm: DCOP Extensions
- 12:30pm: Applications
- 12:50pm: Challenges and Open Questions
- 1:00pm: Done! Lunch? :)
LIL’ BIT OF SHAMELESS PROMOTION :) 

- Tutorial materials are based on our recent JAIR survey paper:


- Includes more models, algorithms, and applications.
- Also available on arXiv.
PRELIMINARIES

AAAI-18 Tutorial on
Multi-Agent Distributed Constrained Optimization
MOTIVATING DOMAIN: SENSOR NETWORK
MOTIVATING DOMAIN:
SENSOR NETWORK
MOTIVATING DOMAIN: SENSOR NETWORK
MOTIVATING DOMAIN: SENSOR NETWORK
MOTIVATING DOMAIN: SENSOR NETWORK
MOTIVATING DOMAIN: SENSOR NETWORK

Model the problem as a CSP
MOTIVATING DOMAIN: SENSOR NETWORK

Model the problem as a CSP
CSP
CONSTRAINT SATISFACTION

• Variables \( X = \{x_1, \ldots, x_n\} \)
• Domains \( D = \{D_1, \ldots, D_n\} \)
• Constraints \( C = \{c_1, \ldots, c_m\} \)
  
  where a constraint \( c_i \subseteq D_{i_1} \times D_{i_2} \times \ldots \times D_{i_n} \)
  
  denotes the possible valid joint assignments for the variables \( x_{i_1}, x_{i_2}, \ldots, x_{i_n} \) it involves

• **GOAL**: Find an assignment to all variables that satisfies all the constraints
CSP
CONSTRAINT SATISFACTION

Model the problem as a CSP

<table>
<thead>
<tr>
<th>x_1</th>
<th>x_3</th>
<th>x_5</th>
<th>Sat?</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>X</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>E</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>S</td>
<td>E</td>
<td>W</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>W</td>
<td>X</td>
</tr>
</tbody>
</table>
MAX-CSP

MAX CONSTRAINT SATISFACTION

Model the problem as a Max-CSP

<table>
<thead>
<tr>
<th>x1</th>
<th>x3</th>
<th>x5</th>
<th>Sat?</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>X</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>E</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>E</td>
<td>W</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>W</td>
<td>W</td>
<td>X</td>
</tr>
</tbody>
</table>
MAX-CSP
MAX CONSTRAINT SATISFACTION

- **Variables** \( X = \{x_1, \ldots, x_n\} \)
- **Domains** \( D = \{D_1, \ldots, D_n\} \)
- **Constraints** \( C = \{c_1, \ldots, c_m\} \)

where a constraint \( c_i \subseteq D_{i_1} \times D_{i_2} \times \ldots \times D_{i_n} \) denotes the possible valid joint assignments for the variables \( x_{i_1}, x_{i_2}, \ldots, x_{i_n} \) it involves

- **GOAL**: Find an assignment to all variables that satisfies a maximum number of constraints
MAX-CSP
MAX CONSTRAINT SATISFACTION

Model the problem as a Max-CSP
Model the problem as a COP
WCSP (COP)  
CONSTRAINT OPTIMIZATION

- Variables $X = \{x_1, \ldots, x_n\}$
- Domains $D = \{D_1, \ldots, D_n\}$
- Constraints $C = \{c_1, \ldots, c_m\}$

where a constraint $c_i : D_{i_1} \times \ldots \times D_{i_n} \rightarrow \mathbb{R}_+ \cup \{\infty\}$ expresses the degree of constraint violation.

**GOAL:** Find an assignment that minimizes the sum of the costs of all the constraints.
WCSP (COP)
CONSTRAINT OPTIMIZATION

CSP ➔ Max-CSP

- Objective: maximize #constraints satisfied
WCSP (COP) CONSTRAINT OPTIMIZATION

- CSP
- COP
- Max-CSP

- Hard constraints to Soft constraints
- Objective: minimize cost
- Objective: maximize #constraints satisfied
Imagine that each sensor is an autonomous agent.

How should this problem be modeled and solved in a decentralized manner?
MULTI-AGENT SYSTEMS

- **Agent**: An entity that behaves autonomously in the pursuit of goals
- **Multi-agent system**: A system of multiple interacting agents

An agent is:
- **Autonomous**: Is of full control of itself
- **Interactive**: May communicate with other agents
- **Reactive**: Responds to changes in the environment or requests by other agents
- **Proactive**: Takes initiatives to achieve its goals
# MULTI-AGENT SYSTEMS

<table>
<thead>
<tr>
<th>Element</th>
<th>Characterization</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Agent</strong></td>
<td></td>
</tr>
<tr>
<td>behavior</td>
<td>deterministic / stochastic</td>
</tr>
<tr>
<td>knowledge</td>
<td>total / partial</td>
</tr>
<tr>
<td>teamwork</td>
<td>cooperative / competitive</td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td></td>
</tr>
<tr>
<td>behavior</td>
<td>deterministic / stochastic</td>
</tr>
<tr>
<td>evolution</td>
<td>static / dynamic</td>
</tr>
</tbody>
</table>
MULTI-AGENT SYSTEMS

Constraint Programming

Decision Theory

Game Theory

DCOP

Dec-MDP; Dec-POMDP

Auctions; Games
Imagine that each sensor is an autonomous agent.

How should this problem be modeled and solved in a decentralized manner?
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

- Agents \( A = \{a_i, \ldots, a_n\} \)
- Variables \( X = \{x_1, \ldots, x_n\} \)
- Domains \( D = \{D_1, \ldots, D_n\} \)
- Constraints \( C = \{c_1, \ldots, c_m\} \)
- Mapping of variables to agents

- **GOAL**: Find an assignment that minimizes the sum of the costs of all the constraints
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

CSP
Max-CSP
COP

- Hard constraints to Soft constraints
- Objective: minimize cost
- Objective: maximize #constraints satisfied
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

- Variables are controlled by agents
- Communication model
- Local agents’ knowledge
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

• Why distributed models?
  • Natural mapping for multi-agent systems
  • Potentially faster by exploiting parallelism
  • Potentially more robust: no single point of failure, no single network bottleneck
  • Maintains more private information
  • ...

DCOP ALGORITHMS

AAAI-18 Tutorial on
Multi-Agent Distributed Constrained Optimization
DCOP ALGORITHMS

- **Complete**
  - Partially Decentralized
    - Synchronous
      - Search
      - Inference
  - Fully Decentralized
    - Synchronous
      - Search
      - Inference
    - Asynchronous
      - Search
      - Inference

- **Incomplete**
  - Fully Decentralized
    - Synchronous
      - Sampling
    - Asynchronous
      - Search
      - Inference
DCOP ALGORITHMS

- Important Metrics:
  - Agent complexity
  - Network loads
  - Message size

Complete

- Partially Decentralized
  - Synchronous
    - Search
    - Inference

- Fully Decentralized
  - Synchronous
    - Search
    - Inference
  - Asynchronous
    - Search

Incomplete

- Fully Decentralized
  - Synchronous
    - Sampling
  - Asynchronous
    - Search
    - Inference
DCOP ALGORITHMS

- Important Metrics:
  - Agent complexity
  - Network loads
  - Message size
  - Anytime
  - Quality guarantees
  - Execution time vs. solution quality

Complete

- Partially Decentralized
  - Synchronous
    - Search
    - Inference
  - Asynchronous
    - Search

Fully Decentralized

- Synchronous
  - Search
  - Inference
  - Sampling

Incomplete

- Fully Decentralized
  - Synchronous
    - Search
  - Asynchronous
    - Inference

DCOP ALGORITHMS

- Systematic process, divided in steps.
- Each agent waits for particular messages before acting.
- Consistent view of the search process.
- Typically, increases idle-time.
DCOP ALGORITHMS

- Decision based on agents’ local state
- Agents’ actions do not depend on sequence of received messages
- Minimizes idle-time
- No guarantees on validity of local views

Complete
- Partially Decentralized
- Synchronous
  - Search
  - Inference

Incomplete
- Fully Decentralized
- Asynchronous
  - Search
  - Inference
DCOP ALGORITHMS

- **Complete**
  - Partially Decentralized
  - Synchronous
    - Search
    - Inference
  - Asynchronous
    - Search
- **Incomplete**
  - Fully Decentralized
  - Synchronous
    - Search
    - Inference
  - Asynchronous
    - Search
    - Inference

**Synchronous Branch and Bound (SBB)**
SBB

Katsutoshi Hirayama, Makoto Yokoo: Distributed Partial Constraint Satisfaction Problem. CP 1997: 222-236
How do we solve this distributedly?

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>$x_j$</th>
<th>Cost ($A,B$)</th>
<th>Cost ($A,C$)</th>
<th>Cost ($B,C$)</th>
<th>Cost ($B,D$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
• Agents operate on a complete ordering
• Agents exchange CPA messages containing partial assignments.
• When a solution is found a LB is broadcasted to all agents.
• The LB is used for branch pruning.
SBB

UB = infinity

A
B
C
D
SBB

UB = infinity
SBB

UB = infinity
SBB

UB = 18
SBB

UB = 18
SBB

UB = 18
SBB

UB = 18
SBB

UB = 18

A

B

C

D

18

23

19

15

8

5

0

18
SBB

UB = 18

A

B

C

D

AAAI-18 Tutorials

Fioretto, Yeoh, Zivan
SBB

UB = 18
SBB

UB = 18
### SBB

<table>
<thead>
<tr>
<th></th>
<th>SBB</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Correct</strong></td>
<td>Yes</td>
</tr>
<tr>
<td>the solution it finds is optimal</td>
<td></td>
</tr>
<tr>
<td><strong>Complete</strong></td>
<td>Yes</td>
</tr>
<tr>
<td>it terminates</td>
<td></td>
</tr>
<tr>
<td><strong>Message Complexity</strong></td>
<td>$O(d)$</td>
</tr>
<tr>
<td>max size of a message</td>
<td></td>
</tr>
<tr>
<td><strong>Network Load</strong></td>
<td>$O(b^d)$</td>
</tr>
<tr>
<td>max number of messages</td>
<td></td>
</tr>
<tr>
<td><strong>Runtime</strong></td>
<td>$O(b^d)$</td>
</tr>
<tr>
<td>max number of cycles</td>
<td></td>
</tr>
</tbody>
</table>

branching factor = $b$
num variables = $d$
Can we speed this up by parallelizing some computations?

*Hint: Are there independent or conditionally independent subproblems?*
These computations are the same; independent of $C!$
Definition: A spanning tree of the constraint graph such that no two nodes in sibling subtrees share a constraint in the constraint graph.
DCOP ALGORITHMS

Distributed Pseudotree Optimization Procedure (DPOP)
DPOP

• Extension of the Bucket Elimination (BE)
• Agents operate on a pseudo-tree ordering
• UTIL phase: Leaves to root
• VALUE phase: Root to leaves

DPOP

Pseudo-tree Ordering

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>(B,D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>8</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>3</td>
</tr>
</tbody>
</table>
DPOP

Pseudo-tree Ordering

\[
\begin{array}{c|c|c}
B & D & (B,D) \\
\hline
r & r & 3 \\
r & g & 8 \\
g & r & 10 \\
g & g & 3 \\
\end{array}
\]

\[
\begin{align*}
\text{MSG to B} & : \\
B & | \text{cost} \\
r & 3 \\
g & 3 \\
\end{align*}
\]

\[
\text{min}\{3, 8\} = 3 \\
\text{min}\{10, 3\} = 3
\]
DPOP

Pseudo-tree Ordering

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>(B,C)</th>
<th>(A,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

AAAI-18 Tutorials

Fioretto, Yeoh, Zivan
DPOP

Pseudo-tree Ordering

MSG to B

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>(B,C)</th>
<th>(A,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

| A | B | | |
|---|---|---|
| r | r | 10 |
| r | g | 8  |
| g | r | 7  |
| g | g | 6  |
DPOP

Pseudo-tree Ordering

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(A,B)</th>
<th>Util C</th>
<th>Util D</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>8</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>20</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>3</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>
DPOP

Pseudo-tree Ordering

MSG to A

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(A,B)</th>
<th>Util C</th>
<th>Util D</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>8</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>20</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>3</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>
DPOP

<table>
<thead>
<tr>
<th>A</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>18</td>
</tr>
<tr>
<td>g</td>
<td>12</td>
</tr>
</tbody>
</table>

optimal cost = 12
DPOP

- Select value for $A = 'g'$
- Send MSG $A = 'g'$ to agents B and C

<table>
<thead>
<tr>
<th>A</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>18</td>
</tr>
<tr>
<td>g</td>
<td>12</td>
</tr>
</tbody>
</table>
DPOP

Pseudo-tree Ordering

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(A,B)</th>
<th>Util C</th>
<th>Util D</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>8</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>20</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>3</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

- Select value for B = ‘g’
- Send MSG B = ‘g’ to agents C and D
DPOP

Pseudo-tree Ordering

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>(B,C)</th>
<th>(A,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>r</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>g</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>r</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>g</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

• Select value for C = ‘g’
DPOP

Pseudo-tree Ordering

VALUE

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>(B,D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>r</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>8</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>3</td>
</tr>
</tbody>
</table>

• Select value for D = ‘g’
# DPOP

<table>
<thead>
<tr>
<th></th>
<th>SBB</th>
<th>DPOP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Correct</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>the solution it finds is optimal</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Complete</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>it terminates</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Message Complexity</strong></td>
<td>O(d)</td>
<td>O($b^d$)</td>
</tr>
<tr>
<td>max size of a message</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Network Load</strong></td>
<td>O($b^d$)</td>
<td>O(d)</td>
</tr>
<tr>
<td>max number of messages</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Runtime</strong></td>
<td>O($b^d$)</td>
<td>O($b^d$)</td>
</tr>
<tr>
<td>max number of cycles</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

branching factor = $b$
num variables = $d$
CRITICAL OVERVIEW

Search Algorithms

- increasing memory
- polynomial
- exponential

Inference Algorithms

- decreasing network load
- exponential
- polynomial
DCOP ALGORITHMS

- Complete
  - Partially Decentralized
    - Synchronous
      - Search
      - Inference
    - Asynchronous
      - Search
      - Inference
  - Fully Decentralized
    - Synchronous
      - Search
      - Inference
    - Asynchronous
      - Search
      - Inference

Incomplete

- Fully Decentralized
  - Synchronous
    - Sampling
  - Asynchronous
    - Search
    - Inference

Distributed Local Search
LOCAL SEARCH ALGORITHMS

• DSA: Distributed Stochastic Algorithm
• MGM: Maximum Gain Messages Algorithm

• Every agent individually decides whether to change its value or not
• Decision involves
  • knowing neighbors’ values
  • calculation of utility gain by changing values
  • probabilities

DSA ALGORITHM

- All agents execute the following
  - Randomly choose a value
  - while (termination is not met)
    - if (a new value is assigned)
      - send the new value to neighbors
    - collect neighbors’ new values if any
  - select and assign the next value based on assignment rule

DSA ALGORITHM

\[
\begin{align*}
\text{Utility}(A,B) &= 5 \\
\text{Utility}(B,C) &= 0 \\
\text{Utility}(C,A) &= 8
\end{align*}
\]
**DSA ALGORITHM**

- **U=10, Δ=10**
- **U=0, Δ=0**
- **U=0, Δ=0**
- **Δ=0**
- **U=8, Δ=0**
- **U=8, Δ=0**

<table>
<thead>
<tr>
<th>Xi</th>
<th>Xj</th>
<th>Utility (A,B)</th>
<th>Utility (B,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
**DSA ALGORITHM**

- $U = 10, \Delta = 2$
- $U = 8, \Delta = 0$
- $U = 8, \Delta = 8$
- $U = 0, \Delta = 0$

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>$x_j$</th>
<th>Utility $(A, B)$</th>
<th>Utility $(B, C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
**DSA ALGORITHM**

- **U=0, Δ=0**
- **U=16, Δ=16**
- **U=5, Δ=5**
- **U=0, Δ=0**

<table>
<thead>
<tr>
<th>X_i</th>
<th>X_j</th>
<th>Utility (A,B)</th>
<th>Utility (B,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

### Notes
- Δ represents the change in utility.
- Utility values are based on the connections between nodes.
DSA ALGORITHM

\[ U=8 \]

\[ U=0, \Delta=-16 \]

\[ U=16, \Delta=0 \]

\[ U=0, \Delta=-8 \]

\[ U=8, \Delta=0 \]

\[ U=8, \Delta=0 \]

\[ \text{Utility} (A,B) \]

\[ \text{Utility} (B,C) \]

<table>
<thead>
<tr>
<th>( x_i )</th>
<th>( x_j )</th>
<th>Utility (A,B)</th>
<th>Utility (B,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
### DSA ALGORITHM

One possible execution trace

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>$x_j$</th>
<th>Utility $(A,B)$</th>
<th>Utility $(B,C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>U=0</td>
<td>U=0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>U=8</td>
<td>U=0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U=0</td>
<td>U=0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>U=8</td>
<td>U=8</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
MGM ALGORITHM

• All agents execute the following
  • Randomly choose a value
  • while (termination is not met)
    • if (a new value is assigned)
      • send the new value to neighbors
    • collect neighbors’ new values if any
    • calculate gain and send it to neighbors
    • collect neighbors’ gains
  • if (it has the highest gain among all neighbors)
    • change value to the value that maximizes gain

MGM ALGORITHM

- All agents execute the following:
  - Randomly choose a value
  - while (terminating condition)
    - if (it has the highest gain among all neighbors)
      - send the received value
      - collect neighbors’ new values if any

Great if you need an anytime algorithm
- collect neighbors’ gains
- if (it has the highest gain among all neighbors)
  - change value to the value that maximizes gain

DCOP EXTENSIONS

AAAI-18 Tutorial on
Multi-Agent Distributed Constrained Optimization
Designing a Marketplace for the Trading and Distribution of Energy in the Smart Grid. AAMAS 2015: 1285-1293
PROSUMER ENERGY TRADING

- Prosumers: capable of both generating and consuming resources
- Each prosumer can sell or buy a given amount of power to another prosumer
- Line capacity and flow constraints are required to be satisfied
- Each offer has a desired utility
- Goal: Find a buy/selling assignment that maximizes the actors’ rewards and is feasible with the operating power constraints
PROSUMER ENERGY TRADING

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

\[ f_{ab} \]

<table>
<thead>
<tr>
<th>a</th>
<th>c</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>-2</td>
<td>1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

\[ f_{ac} \]

<table>
<thead>
<tr>
<th>b</th>
<th>c</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

\[ f_{bc} \]

Designing a Marketplace for the Trading and Distribution of Energy in the Smart Grid. AAMAS 2015: 1285-1293
What if Alice cannot disclose the costs associated her action?

What if we want to describe the scenario in which

- Bob desires to gain 0.2 for selling 1 KW of power to Carl

- Car desires to gain 0.1 for buying 1 KW of power from Bob?
ASYMMETRIC DCOP

Constraint Programming

Decision Theory

Game Theory

Asymmetric costs/rewards

DCOP

Dec-MDP; Dec-POMDP

Auctions; Games
Asymmetric DCOPs are DCOPs where:

- A joint assignment may produce different costs for the agents participating in a constraint

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>g</td>
<td>3</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>2</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>10</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>0</td>
</tr>
</tbody>
</table>
ASYMMETRIC DCOP

• Asymmetric DCOPs are DCOPs where:
• A joint assignment may produce different costs for the agents participating in a constraint

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cost A</th>
<th>Cost B</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>g</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>r</td>
<td>g</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>g</td>
<td>r</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
PROSUMER ENERGY TRADING

\[
\begin{array}{c|c|c}
\text{a} & \text{b} & \text{U} \\
0 & 0 & 0.2 \\
0 & -1 & 0 \\
-2 & 1 & 0.2 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{a} & \text{c} & \text{U} \\
0 & 0 & 0.1 \\
0 & 1 & 0.1 \\
-2 & 1 & 0.2 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{a} & \text{b} & \text{U} \\
0 & 0 & 0.1 \\
0 & -1 & 0.1 \\
-2 & 1 & 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{a} & \text{c} & \text{U} \\
0 & 0 & 1 \\
0 & 1 & 0 \\
-2 & 1 & 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{a} & \text{c} & \text{U} \\
0 & 0 & 0 \\
-1 & 1 & 0.2 \\
-2 & 1 & 0.1 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{a} & \text{c} & \text{U} \\
0 & 0 & 0 \\
-1 & 1 & 0.1 \\
-2 & 1 & 0.2 \\
\end{array}
\]
ASYMMETRIC DCOP

• Why asymmetric DCOPs?
  • ...
ASYMMETRIC DCOP

• Why asymmetric DCOPs?
  • Models richer forms of cooperation
  • Privacy: Agents do not need to reveal the costs associated to their action
• Resource allocation problems:
  • Different costs for using the same resource
  • Different preferences
PROSUMER ENERGY TRADING

• What if a new prosumer would like to join the market?

• What if a prosumer would like to modify her preferences?
PROSUMER ENERGY TRADING

\[
\begin{array}{c|c|c}
\text{a} & \text{b} & \text{U} \\
0 & 0 & 0.3 \\
0 & -1 & 0 \\
-2 & 1 & 0.2 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{a} & \text{c} & \text{U} \\
0 & 0 & 0.1 \\
0 & 1 & 0.2 \\
-2 & 1 & 0.2 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c}
\text{b} & \text{c} & \text{U} \\
0 & 0 & 0 \\
-1 & 1 & 0.2 \\
1 & 1 & 0.3 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{a} & \text{d} & \text{U} \\
1 & 0 & 0.1 \\
2 & -1 & 0.4 \\
-2 & 1 & 0.2 \\
\end{array}
\]
DYNAMIC DCOP

Dynamic environment

Constraint Programming

DCOP

Decision Theory

Dec-MDP; Dec-POMDP

Game Theory

Auctions; Games
DYNAMIC DCOP

- A Dynamic DCOP is sequence \( P_1, P_2, \ldots, P_k \) of \( k \) DCOPs
- The agent knowledge about the environment is confined within each time step
- Each DCOP is solved sequentially
DYNAMIC DCOP

• Why dynamic DCOPs?
  • ...
DYNAMIC DCOP

• Why dynamic DCOPs?
  • MAS commonly exhibit dynamic environments
  • The capture scenarios with:
    • Moving agents, change of constraints, change of preferences
    • Additional information become available during problem solving
  • Application domains: Sensor networks, cloud computing, smart home automation, …
APPLICATIONS

AAAI-18 Tutorial on
Multi-Agent Distributed Constrained Optimization
DCOP APPLICATIONS

• Scheduling Problems
  • Taking DCOP to the Real World: Efficient Complete Solutions for Distributed Multi-Event Scheduling. AAMAS 2004

• Radio Frequency Allocation Problems
  • Improving DPOP with Branch Consistency for Solving Distributed Constraint Optimization Problems. CP 2014

• Sensor Networks
  • Preprocessing techniques for accelerating the DCOP algorithm ADOPT. AAMAS 2005

• Home Automation
  • A Multiagent System Approach to Scheduling Devices in Smart Homes. AAMAS 2017, IJCAI 2016

• Traffic Light Synchronization
  • Evaluating the performance of DCOP algorithms in a real world, dynamic problem. AAMAS 2008

• Disaster Evacuation
  • Disaster Evacuation Support. AAAI 2007; JAIR 2017

• Combinatorial Auction Winner Determination
  • H-DPOP: Using Hard Constraints for Search Space Pruning in DCOP. AAAI 2008
MEETING SCHEDULING

- Meeting 1: Alice, Bob, Carl
- Meeting 2: Bob, Carl
- ...

- Alice is only free in the mornings from 9am-noon
- Bob prefers to not meet during lunch (noon-1pm)
- Carl does not wake up until 11am and loves late evening meetings
- ...

MEETING SCHEDULING

- Values: time slots to hold the meetings
- All agents participating in a meeting must meet at the same time
- All meetings of an agent must occur at different times

TRAFFIC FLOW CONTROL

- Given a set of traffic lights in adjacent intersections
- How coordinate them to create green waves?

---


Junges, R., & Bazzan, A. L. Evaluating the performance of DCOP algorithms in a real world, dynamic problem. AAMAS 2008: 463-470
TRAFFIC FLOW CONTROL

- Agents: Each traffic light
- Values: Flow traffic direction

Junges, R., & Bazzan, A. L. Evaluating the performance of DCOP algorithms in a real world, dynamic problem. AAMAS 2008: 463-470
TRAFFIC FLOW CONTROL

- Agents: Each traffic light
- Values: Flow traffic direction
- Conflict if 2 neighboring signals choose different directions

---

\[
x_1 \quad x_2 \quad x_3
\]

- N/S
- W/E

---


TRAFFIC FLOW CONTROL

• Cost functions model the number of incoming vehicles

• Maximize the traffic flow

Cost functions model the number of incoming vehicles.

Maximize the traffic flow.


Junges, R., & Bazzan, A. L. Evaluating the performance of DCOP algorithms in a real world, dynamic problem. AAMAS 2008: 463-470
SMART DEVICES
HOME ASSISTANTS
The SBDS problem is the problem of scheduling the devices in smart buildings. It involves the scheduling of devices in a multi-building environment, considering constraints and objectives such as energy costs, user comfort, and operational efficiency.

We now provide a description of the definitions:

- **Definition 1**: Let \( I = \{ x_1, x_2, \ldots, x_n \} \) be the set of its variables. The goal is to find an optimal complete solution to represent the set of constraints.

- **Definition 2**: Let \( F = (I, X) \) be a subgraph of the constraint graph, where \( X \) is the set of all variables included in the solution.

- **Definition 3**: The SBDS problem is also subject to constraints that define the time horizon of the scheduling process. We use \( H \) to denote the set of time steps in the horizon.

- **Definition 4**: A solution is said to be valid if it satisfies all the constraints associated with the pricing schema adopted, which expresses the cost of the devices based on the state of the device and the price function.

We now provide a description of the multi-objective approaches in Section 6. An SBDS problem is considered as a multi-objective optimization problem, where the objective is to minimize the total cost of all the applicable constraints in the system.

A **smart home** has:

- **Smart devices** (roomba, HVAC) that it can control
- **Sensors** (cleanliness, temperature)
- A set of locations

Smart device:
- A set of actions it can perform (clean, charge)
- Power consumption associated to each action.

Scheduling Rules:

`living_room_cleanliness ≥ 75 before 1800`
Goal

Cleanliness (%)

Living_room cleanliness ≥ 75 before 1800

Device Schedule

Cleanliness (%)

Battery Charge (%)

Goal

75

60

45

30

15

0

Time

1400

1500

1600

1700

1800

Deadline

75

60

45

30

15

0

DEVICE SCHEDULING

To ensure that a goal state can be achieved across the device scheduling, feasibility of schedules must hold for all time steps. Additionally, a rule is associated with a sensor interval. Each scheduling rule indicates a goal state (either a desired condition or a specified property).

Definition 3 (Predicted State Trajectory)

A schedule is a sequence of states that are feasible according to the scheduling rules. The ASR of Equation (1) is illustrated in Figure 2 by dotting the states that are feasible at each time step. Thus:

$$ p_t \rightarrow R \rightarrow \{ v \} $$

where $R$ is the set of states that are feasible at time step $t$, and $\{ v \}$ represents a goal state property. To denote the set of states that are feasible across all time steps, we use $F$. The feasibility of schedules is critical for ensuring that the goal state can be achieved across all time steps.

/* Insert diagram image */
How to schedule smart devices to satisfy the user preferences while
1) minimizing energy costs and
2) reducing peaks in load demand?

Assumptions: Each home have communication and controllable load capabilities.
SMALL HOMES
DEVICE SCHEDULING

SMALL HOMES DEVICE SCHEDULING

### Domains
- **Action** | **State property** | **Power (kW/h)**
- run | cleanliness, battery charge | 0.0
- charge | battery charge | 0.26
- stop | | 0.0

### Environmental variables

### User Preferences (constraints)
- living_room cleanliness ≥ 75 before 1800

Figure 1: Example DCOP
SMART HOMES
DEVICE SCHEDULING

Figure 1(a) shows the constraint graph of a sample DCOP $G$. The goal is to find an optimal complete solution $\phi$ that is consistent with the variables' domains. The domains are $\mathcal{X} = \{1, 2, 3\}$ for agent $i$. For each agent $i$, the following functions are defined:

- $\mathcal{F}_i(a) = \{1, 2, 3\}$
- $\mathcal{L}_i(x) = \{1, 2, 3\}$
- $\mathcal{H}_i(x) = \{1, 2, 3\}$
- $\mathcal{E}_i(x) = \{1, 2, 3\}$
- $\mathcal{D}_i(x) = \{1, 2, 3\}$

We now provide a description of the Objective (soft constraints):

- **Objective**: $\sum_{i \in \mathcal{F}_i(a)} \mathcal{H}_i(x)\mathcal{D}_i(x) + \sum_{i \in \mathcal{L}_i(x)} \mathcal{H}_i(x)\mathcal{D}_i(x) + \sum_{i \in \mathcal{H}_i(x)} \mathcal{F}_i(a)\mathcal{L}_i(x) + \sum_{i \in \mathcal{E}_i(x)} \mathcal{F}_i(a)\mathcal{L}_i(x) + \sum_{i \in \mathcal{D}_i(x)} \mathcal{F}_i(a)\mathcal{L}_i(x)$

The real-time energy price schema is shown in the figure. The price fluctuates at different times of the day, with some examples shown: $8:00, 9:00, 10:00, 11:00, \ldots$.

CHALLENGES AND OPEN QUESTIONS

AAAI-18 Tutorial on
Multi-Agent Distributed Constrained Optimization
MAS DECISION MAKING

• Decentralized decision making difficult due to:
  • Large number of interacting entities
MAS DECISION MAKING

• Decentralized decision making difficult due to:
  • Large number of interacting entities
  • Can we use **decomposition techniques** to reduce the amount of interactions?
  • Can we create **hierarchical models** to increase parallelism and efficiency?

- Ferdinando Fioretto, Hong Xu, Sven Koenig, TK Satish Kumar. "Constraint Composite Graph-Based Lifted Message Passing for Distributed Constraint Optimization Problems". In International Symposium on Artificial Intelligence and Mathematics (ISAIM), 2018.
MAS DECISION MAKING

- Decentralized decision making difficult due to:
  - Large number of interacting entities
  - Can we use **decomposition techniques** to reduce the amount of interactions?
  - Can we create **hierarchical models** to increase parallelism and efficiency?

---

- Bistaffa, Farinelli, Bombieri. "Optimising memory management for belief propagation in junction trees using GPGPUs “, ICPADS 2014
- Ferdinando Fioretto, Hong Xu, Sven Koenig,TK Satish Kumar. "Constraint Composite Graph-Based Lifted Message Passing for Distributed Constraint Optimization Problems". In International Symposium on Artificial Intelligence and Mathematics (ISAIM), 2018.
DYNAMIC ENVIRONMENT

- Interaction in a dynamic environment is required to be robust to several changes

DYNAMIC ENVIRONMENT

• Interaction in a dynamic environment is required to be robust to several changes
  • How do agents respond to dynamic changes?
  • Can we study adaptive algorithms so that the MAS interaction is resilient and adaptive to changes in the communication layer, the underlying constraint graph, etc.?

AGENT PREFERENCES

- How to model, learn, and update agent preferences?
AGENT PREFERENCES

• How to model, learn, and update agent preferences?
  • Agent’s preferences are assumed to be available. This is not always feasible. How to efficiently elicit agents’ preferences?
  • When full elicitation is not possible, how to adaptively learn the preference of an agent?

THANK YOU!

Ferdinando Fioretto, William Yeoh, Roie Zivan