A Local Search Based Approach to Solve Continuous DCOPs

Amit Sarker, Moumita Choudhury, and Md. Mosaddek Khan.

University of Dhaka, Bangladesh

20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)

AAMAS 2021 LONDON, UK

Introduction

Distributed Constraint Optimization Problems (DCOPs) are a powerful framework to model cooperative Multi-agent Systems. This framework has been applied to various areas of multi-agent coordination, such as:

- Distributed meeting scheduling.
- Sensor networks.
- Smart grids.

and many more.

Definition of a DCOP

A DCOP is defined as a tuple $\langle A, X, D, F, \alpha \rangle$, where,

- A $= \{a_1, a_2, ..., a_n\}$ is a finite set of agents.
- X $= \{x_1, x_2, ..., x_m\}$ is a finite set of discrete decision variables.
- D $= \{D_1, D_2, ..., D_m\}$ is a set of finite discrete domains where each D_i corresponds to the domain of variable x_i.
- F $= \{f_1, f_2, ..., f_k\}$ is a finite set of cost functions, with each $f_i : \prod_{x_j \in x^i} D_j \rightarrow R$ defined over a set of variables $x^i \subseteq X$.
- $\alpha : X \rightarrow A$ is a mapping function, which associates each variable $x_i \in X$ to an agent $a_i \in A$.

Objective of a DCOP:

$$X^* = \underset{X}{\operatorname{argmin}} \sum_{i=1}^{k} f_i(x^i) \tag{1}$$

Continuous DCOPs (C-DCOPs)

The traditional DCOP model is based on an assumption; that is, the variables are discrete decision variables. Nevertheless, a number of applications can be best modeled with continuous valued variables, such as:

- Target tracking sensor orientation.
- Cooperative air and ground surveillance.
- Network coverage using low duty-cycled sensors.

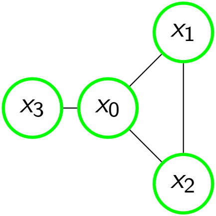
and many more.

Definition of a C-DCOP

A Continuous DCOP (C-DCOP) can also be described by a tuple $\langle A, X, D, F, \alpha \rangle$, where A, F, and α are exactly the same as those in a DCOP. X and D are defined as follows:

- X $= \{x_1, x_2, ..., x_m\}$ is a finite set of continuous decision variables.
- D $= \{D_1, D_2, ..., D_m\}$ is a set of continuous domains. Each variable x_i can choose any value from a range, $D_i = [LB_i, UB_i]$.

Example of a C-DCOP

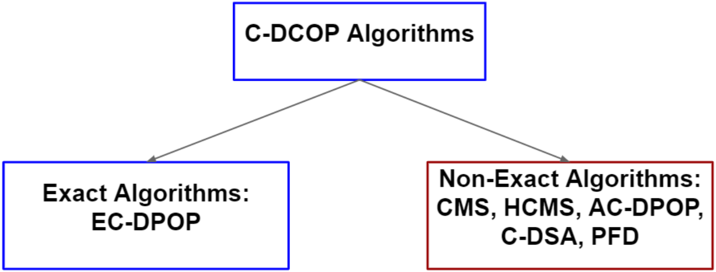


(a) Constraint Graph

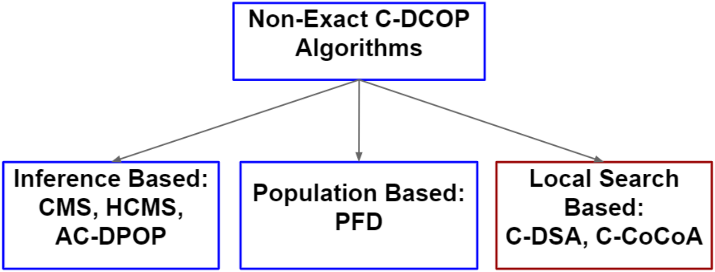
$$\forall x_i \in X : D_i = [-20, 20]$$
$$f(x_0, x_1) = x_0^2 - 2x_0x_1 + 2x_1^2$$
$$f(x_0, x_2) = x_0x_2 + 3x_2^2$$
$$f(x_0, x_3) = x_0x_3 + x_3^2$$
$$f(x_1, x_2) = x_1^2 - x_1x_2 + 2x_2^2$$

(b) Cost Functions

Exact & Non-exact C-DCOP Algorithms



Non-exact C-DCOP Algorithms



Overview

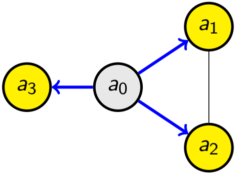
C-CoCoA is a non-iterative algorithm that is able to find high-quality solutions with a smaller communication overhead than the existing state-of-the-art C-DCOP solvers.

- Agents' states used in C-CoCoA:
  - *IDLE*: Initial state for each agent.
  - *ACTIVE*: The agent is now active and working to find an assignment for its variable.
  - *HOLD*: Variable assignment for the agent is delayed and the agent waits for more information from neighbors.
  - *DONE*: The agent has completed the assignment for its variable.

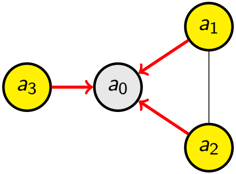
Overview

- Message types used in C-CoCoA:
    - **InquiryMessage**: Sent by an agent to its neighbors at the start of the algorithm.
    - **CostMessage**: Neighbors' reply against the **InquiryMessage**.
    - **UpdateStateMessage**: An agent updates its state.
    - **SetValueMessage**: An agent assigns a value to its variable.

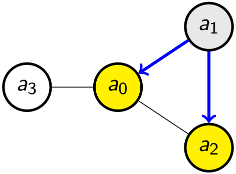
The Algorithm



(a) **InquiryMessage** from a_0



(b) **CostMessage** to a_0



(c) **InquiryMessage** from a_1

- Discretize each continuous domain into d points.
- The algorithm starts by randomly activating an agent a_i.
- a_i sends **InquiryMessage** to all the neighbors $a_j \in \mathcal{N}_i$.

The Algorithm

- Each neighbor a_j calculates the cost by using the following equation:

$$\zeta_{j,k} = \min_{x_{j,l} \in D_j} \sum_{C \in F_j} C(\widetilde{x_j} \cap x_{i,k} \cap x_{j,l}) \tag{2}$$

- Agent a_j then sends the cost map to the inquiring agent a_i.
- a_i then finds the value of its variable by using the following equation:

$$\delta = min \sum_{j=1}^{|\mathcal{N}_i|} \zeta_{j,k}; \qquad \rho = \{k : \sum_{j=1}^{|\mathcal{N}_i|} \zeta_{j,k} = \delta\} \tag{3}$$

The Algorithm

- This assignment is near-optimal within the discretized domain. In order to find the best solution within the actual continuous domain, we use a non-linear optimization technique.
- For employing the gradient-based non-linear optimization, agent a_i calculates the local objective function $F_{\mathcal{N}_i}^{a_i}$ by using the following equation:

$$F_{\mathcal{N}_i}^{a_i} = \sum_{a_j \in \mathcal{N}_i} f(a_i, a_j) \tag{4}$$

- Specifically, the agent a_i minimizes the local objective function $F_{\mathcal{N}_i}^{a_i}$ and updates the value v_x of each variable $x \in x_{\mathcal{N}_i}^{a_i}$ according to the following equation:

$$v_x(t) = v_x(t-1) - \alpha \frac{\partial F_{\mathcal{N}_i}^{a_i}}{\partial x_{\mathcal{N}_i}^{a_i}} \Bigg|_{\operatorname{argmin}_{x_i} F_{\mathcal{N}_i}^{a_i}(x_{\mathcal{N}_i}^{a_i})}^{v_x} \tag{5}$$

The Algorithm

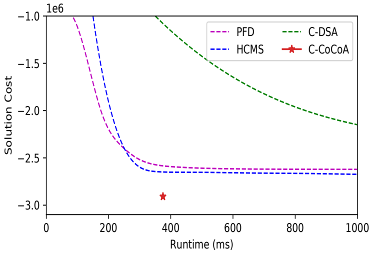
- The agent continues this update process until it converges or a maximum number of iterations is reached. After termination, the current value of v_x is actually the approximate optimal assignment for the variable x_i.
- The agent a_i then updates its state to *DONE* and communicates to its neighbors $a_j \in \mathcal{N}_i$ in a **SetValueMessage**. By receiving this message, the neighbors update their *CPA* with the value of x_i and trigger the algorithm for them.
- Note that each agent can only assign its value once and when assigned it cannot change its value. Thus C-CoCoA is a non-iterative approach.

Theoretical Analysis

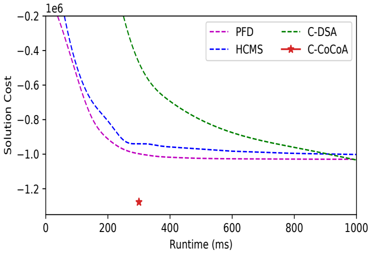
In a binary constraint graph $G = (N, E)$, in the worst case:

- The total number of messages sent or received by an agent a_i is $5|N| + d|N|$.
- The total message size for an agent a_i is $O(2|N|^2 + d|N|)$.
- The overall computational complexity is $O(|N|(d^2 + b))$.

Random C-DCOPs



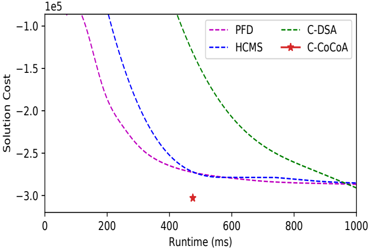
(a) Dense Graphs



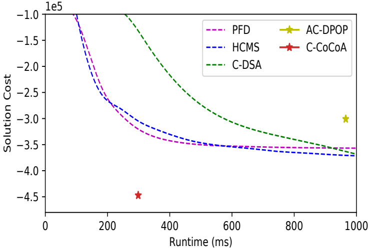
(b) Sparse Graphs

- Random C-DCOPs with 50 agents and graph density 0.2 to 0.6.
- C-CoCoA outperforms state-of-the-art non-exact algorithms by 7.97% – 24.58% on dense random C-DCOPs, 18.75% – 25.43% on sparse random C-DCOPs.

Random C-DCOPs



(a) Scale-free Graphs



(b) Random Tree

- For scale-free network, we use 100 agents.
- C-CoCoA outperforms state-of-the-art non-exact algorithms by 2.63% – 5.94% on scale-free graphs, 11.31% – 21.42% on random trees.

Random Graphs

Table: Solution quality of C-CoCoA, PFD, HCMS, and C-DSA on random graphs with varying number of agents.

		C-CoCoA	PFD	HCMS	C-DSA
$\lvert A \rvert = 30$	$p = 0.2$	-554,339	-469,093	-423,383	-493,730
	$p = 0.6$	-1,251,118	-1,042,611	-1,055,878	-1,096,852
$\lvert A \rvert = 50$	$p = 0.2$	-1,311,988	-1,030,328	-974,416	-1,118,344
	$p = 0.6$	-3,086,005	-2,623,534	-2,730,943	-2,334,385
$\lvert A \rvert = 70$	$p = 0.2$	-2,253,114	-1,858,646	-2,049,757	-1,636,508
	$p = 0.6$	-5,486,958	-4,494,102	-5,060,671	—

Conclusions & Future Work

- The classical DCOP model deals with discrete variables. But this assumption of the variables being discrete is not applicable to many real-world problems.
- In this paper, we propose a local search based algorithm, C-CoCoA, that is able to solve DCOPs with continuous variable.
- Basically, we show that a simple extension of a local search algorithm can find better solution in significantly less time than the complex inference based algorithms.
- In future, we would like to further investigate the potential of C-CoCoA on various C-DCOP applications. We would also like to explore the ways to extend C-CoCoA to solve multi-objective and asymmetric C-DCOPs.

Thank you for watching the presentation.