C-CoCoA: A Continuous Cooperative Constraint Approximation Algorithm to Solve Functional DCOPs

Extended Abstract

Amit Sarker, Abdullahil Baki Arif, Moumita Choudhury, and Md. Mosaddek Khan
Department of Computer Science and Engineering, University of Dhaka, Dhaka, Bangladesh
{amitcsedu99,ahb.arif,moumitach22}@gmail.com,mosaddek@du.ac.bd

ABSTRACT

Distributed Constraint Optimization Problems (DCOPs) are a suitable formulation for coordinating interactions (i.e. constraints) in cooperative multi-agent systems. The traditional DCOP model deals with variables that can take only discrete values. However, there are many applications where the variables are continuous decision variables. The existing methods for solving DCOPs with continuous variables come with a huge computation and communication overhead. In this paper, we apply continuous non-linear optimization methods on Cooperative Constraint Approximation (CoCoA) algorithm. Empirical results show that our algorithm is able to provide high-quality solutions at the expense of small communication cost and execution time.

KEYWORDS

Distributed Problem Solving; DCOPs; Functional DCOPs

ACM Reference Format:
Amit Sarker, Abdullahil Baki Arif, Moumita Choudhury, and Md. Mosaddek Khan. 2020. C-CoCoA: A Continuous Cooperative Constraint Approximation Algorithm to Solve Functional DCOPs: Extended Abstract. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION AND BACKGROUND

Distributed Constraint Optimization Problems (DCOPs) are a powerful framework to model cooperative multi-agent systems wherein multiple agents communicate directly or indirectly with each other. The agents act autonomously in a common environment in order to optimize a global objective which is an aggregation of their corresponding constraint cost functions. Each of the functions is associated with a set of variables controlled by the corresponding agents. In DCOPs, agents need to coordinate value assignments to their variables in such a way that maximize their aggregated utility or minimize the overall cost [4, 13, 14]. A number of multi-agent coordination problems, such as meeting scheduling [12], multi-robot coordination [20] and smart homes [5, 16], have been dealt with this model.

A DCOP is defined as a tuple $\langle \mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F}, \boldsymbol{\alpha} \rangle$, where, $\mathbf{A}$ is a finite set of agents, $\mathbf{X}$ is a finite set of discrete decision variables, $\mathbf{D}$ is a set of finite discrete domains, $\mathbf{F}$ is a finite set of cost functions, $\boldsymbol{\alpha} : \mathbf{X} \rightarrow \mathbf{A}$ is a mapping function. The traditional DCOP model is based on an assumption; that is, each of the variables that are involved in the constraints can take values from discrete domain(s) and a constraint is typically represented in a cost (i.e. utility) table. During almost the last two decades, a number of algorithms [3, 13, 14] and their extensions [10, 11, 15, 21] have been developed to solve DCOPs involving discrete valued variables. Nevertheless, a number of applications, such as target tracking sensor orientation [6], cooperative air and ground surveillance [7], Network coverage using low duty-cycled sensors [9] and many others besides, can be best modeled with continuous-valued variables. Therefore, the traditional DCOP setting is not well-suited to such algorithms. To address this, the regular DCOP model is extended for continuous-valued variables [17]. Later, [8] refer this continuous version of DCOP as Functional DCOPs (F-DCOPs).

In more detail, [17] propose a new version of the Max-Sum algorithm (i.e. Continuous Max-Sum - CMS) in order to solve continuous-valued DCOPs. Then, Hybrid CMS (HCMS) has been proposed which uses discrete Max-Sum as the underlying framework with the addition of a continuous non-linear optimization method [19]. Note that none of CMS and HCMS provides quality guarantees on the solutions because both of them are based on discrete Max-Sum which does not provide any quality guarantees when applied to general graphs [8]. Recently three extensions of the Distributed Pseudo-tree Optimization Procedure (DPOP) [14] algorithm has been proposed – Exact Functional DPOP (EF-DPOP), Approximate Functional DPOP (AF-DPOP) and Clustered AF-DPOP (CAF-DPOP) [8]. However, as they are based on DPOP, a key limitation of these approximate algorithms is that they require exponential memory.

Against this background, we extend the Cooperative Constraint Approximation (CoCoA) [18] algorithm so that it can solve functional DOCPs. Our continuous version of CoCoA, that we call C-CoCoA, is an approximate local search algorithm that can solve F-DCOPs with a very lower communication cost. In C-CoCoA, we combine the discrete CoCoA algorithm with continuous non-linear optimization methods. We empirically show that C-CoCoA outperforms HCMS and AF-DPOP in terms of solution quality, number of messages and time.

2 THE C-COCOA ALGORITHM

The C-CoCoA algorithm first discretize the domain of the variables into a fixed number of points. Note that the states of the participating agents are defined as IDLE, ACTIVE, HOLD and DONE, as defined in [18]. We define a set ψ that contains the set of agents who finish their variable assignments. Therefore, $A - \psi$ is the set of unassigned agents (i.e. the value assignments to their variables are not finished). Then in the initialization step, each agent initializes its state to IDLE, the Current Partial Assignment (CPA) with an empty

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

assignment and ψ with an empty set. After this step, C-CoCoA activates an agent a_i randomly from the set $A - \psi$ as starting with any agent yields the same result. Upon activation, the agent a_i sends an inquiry message to its neighboring agents $\mathcal{N}_i$. When a_i sends the inquiry message to $a_j \in \mathcal{N}_i$, each a_j calculates cost for every value in the discretized domain of a_i using the following equation:

$$\zeta_{j,k} = \min_{x_{j,l} \in D_j} \sum_{C \in F_j} C(\widetilde{x_j} \cap x_{i,k} \cap x_{j,l}) \tag{1}$$

In Equation 1, $\zeta_{j,k}$ is the cost for the k^{th} value of agent a_i's discretized domain which is calculated by the neighbor a_j, $x_{j,l}$ indicates that x_j is assigned the l^{th} value of a_j's discretized domain D_j, C is the cost for the function which is an element of all the constraint function set F_j between agent a_i and a_j, $\widetilde{x_j}$ is the current partial assignment sent from a_i to a_j that contains the known assigned values of the neighbors of a_i, $x_{i,k}$ indicates that x_i is assigned the k^{th} value of agent a_i's discretized domain D_i. Agent a_j calculates $\zeta_{j,k}$ for all the values of $k \in D_i$ and the resulting cost map $\zeta_j = \{\zeta_{j,1}, \zeta_{j,2}, \ldots, \zeta_{j,|D_i|}\}$ is sent to the inquiring agent a_i. Then, a_i finds the value of its variable x_i from the following equation:

$$\delta = min \sum_{j=1}^{|\mathcal{N}_i|} \zeta_{j,k}; \qquad \rho = \{k : \sum_{j=1}^{|\mathcal{N}_i|} \zeta_{j,k} = \delta\} \tag{2}$$

Here, δ is the minimum aggregated cost received from the neighbors for each $k \in D_i$, ρ is a set of values from agent a_i's domain for which the cost is minimum and $\zeta_{j,k}$ is the received cost messages from its neighbors. Agent a_i also stores the variable values of $a_j \in \mathcal{N}_i$ in a set χ for which the minimum cost results.

Now, more than one value in a_i's domain can achieve the minimum cost. In this case, a *unique-first* approach is followed to determine whether the current solution is accepted or not. In this approach, $|\rho|$ is compared with a bound β. The initial value of β is set to 1. This means that the value is acceptable if it is a unique local optimum. If $|\rho| > \beta$, agent a_i goes into HOLD state and waits for more information. Otherwise, a value is selected randomly from ρ and is assigned to its controlled variable. 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. We choose gradient-based optimization approach because this can be implemented in a decentralized way using only local information. Now, for employing the gradient-based nonlinear optimization, agent a_i calculates the local objective function $F^{a_i}_{\mathcal{N}_i}$ by using the following equation:

$$F^{a_i}_{\mathcal{N}_i} = \sum_{a_j \in \mathcal{N}_i} f(a_i, a_j) \tag{3}$$

where, $f(a_i, a_j)$ is the cost function that is related to agent a_i and its direct neighbor $a_j \in \mathcal{N}_i$. After that, the agent a_i performs gradient-based approach for optimizing its local objective function $F^{a_i}_{\mathcal{N}_i}(x^{a_i}_{\mathcal{N}_i})$ where, $x^{a_i}_{\mathcal{N}_i}$ is the set of all the related variables with $F^{a_i}_{\mathcal{N}_i}$. Agent a_i assigns every variable $x \in x^{a_i}_{\mathcal{N}_i}$ with the corresponding value from the previously stored set χ as the initial values in the gradient-based optimization method. Specifically, the agent a_i minimizes the local objective function $F^{a_i}_{\mathcal{N}_i}$ and updates the value v_x of each variable $x \in x^{a_i}_{\mathcal{N}_i}$ according to the following equation:

$$v_x(t) = v_x(t-1) - \alpha \frac{\partial F^{a_i}_{\mathcal{N}_i}}{\partial x^{a_i}_{\mathcal{N}_i}}\Big|^{v_x}_{\text{argmin}_{x_i} F^{a_i}_{\mathcal{N}_i}(x^{a_i}_{\mathcal{N}_i} = v_x)} \tag{4}$$

In Equation 4, α is the *learning rate* of 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. Then the agent a_i updates its state to DONE, updates the set ψ and communicates to its neighbors $a_j \in \mathcal{N}_i$ in a SetValue message. By receiving this message, each neighbor a_j updates its CPA with the value of x_i and repeats the algorithm for the unassigned agents ($a_i \in A - \psi$). When the set $A - \psi$ is empty (all the agents finish their variable assignment), the algorithm terminates. Note that each agent can only assign its value once and when assigned it cannot change its value. To be precise, each agent updates its value locally with gradient descent and sends the setValue() message only once to a neighbor, and thus C-CoCoA is a non-iterative approach.

3 RESULTS AND CONCLUSIONS

When we define the total number of agents $|A| = n$, d is the total number of discrete points taken from the agents' continuous domain, b is the number of times an agent updates the values of the variables in the gradient based update, then in C-CoCoA, the total number of messages sent or received by an agent a_i is $O(5n + dn)$, the total message size for an agent a_i is $O(2n^2 + dn)$ and the overall computational complexity is $O(n(d^2 + b))$. We empirically compare the performance of C-CoCoA with HCMS and AF-DPOP in terms of solution cost (C), time in sec (T) and the number of messages (M). Table 1 shows the detailed comparison. We set the number of agents to 50 and use the Erdős-Rényi topology [2]. Moreover, we stop HCMS after 500 iterations (I). The averages are taken over 50 randomly generated problems.

Table 1: Comparison between C-CoCoA and the competing algorithms in terms of Solution Cost, Time and No. of messages

Graph Type	Algorithm	I	C	T (sec)	M
Sparse	C-CoCoA	N/A	**-1266521.13**	**53.85**	**2510**
	HCMS	500	-1064611.6	66.61	98464
Dense	C-CoCoA	N/A	**-282012.78**	**275.66**	**2603**
	HCMS	500	-260016.83	340.75	519440
Scale-Free	C-CoCoA	N/A	**-713674.37**	**38.95**	725
	HCMS	500	-575538.56	131.51	195040
	AF-DPOP	N/A	-306311.19	185.87	**100**

In all the experimental settings, C-CoCoA shows better results than the other benchmarking algorithms in terms of solution quality, time and number of messages propagated. In the future, we would like to further investigate the potential of C-CoCoA on various F-DCOP applications, as recommended in [1, 8, 17]. We would also like to explore the ways to extend C-CoCoA to solve multi-objective and asymmetric DCOPs having continuous-valued variables.

REFERENCES

[1] M. Choudhury, S. Mahmud, and M. M. Khan. 2020. A Particle Swarm Based Algorithm for Functional Distributed Constraint Optimization Problems. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-20)*.

[2] Paul Erdős and Alfréd Rényi. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* 5, 1 (1960), 17–60.

[3] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings. 2008. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Vol. 2. 639–646.

[4] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. 2018. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research* 61 (2018), 623–698.

[5] Ferdinando Fioretto, William Yeoh, and Enrico Pontelli. 2017. A multiagent system approach to scheduling devices in smart homes. In *Proceedings of the 16th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 981–989.

[6] Stephen Fitzpatrick and L Meetrens. 2003. Distributed Sensor Networks A multiagent perspective, chapter Distributed Coordination through Anarchic Optimization. (2003).

[7] Ben Grocholsky, James Keller, Vijay Kumar, and George Pappas. 2006. Cooperative air and ground surveillance. *IEEE Robotics & Automation Magazine* 13, 3 (2006), 16–25.

[8] Khoi D Hoang, William Yeoh, Makoto Yokoo, and Zinovi Rabinovich. 2019. New Algorithms for Functional Distributed Constraint Optimization Problems. *arXiv preprint arXiv:1905.13275* (2019).

[9] Chih-fan Hsin and Mingyan Liu. 2004. Network coverage using low duty-cycled sensors: random & coordinated sleep algorithms. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*. 433–442.

[10] M. M. Khan, L. Tran-Thanh, and N. R. Jennings. 2018. A generic domain pruning technique for gdl-based dcop algorithms in cooperative multi-agent systems. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 1595–1603.

[11] M. M. Khan, L. Tran-Thanh, W. Yeoh, and N. R. Jennings. 2018. A near-optimal node-to-agent mapping heuristic for gdl-based dcop algorithms in multi-agent systems. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 1613–1621.

[12] Rajiv T Maheswaran, Milind Tambe, Emma Bowring, Jonathan P Pearce, and Pradeep Varakantham. 2004. Taking DCOP to the real world: Efficient complete solutions for distributed multi-event scheduling. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 310–317.

[13] Pragnesh Jay Modi, Wei-Min Shen, Milind Tambe, and Makoto Yokoo. 2005. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence* 161, 1-2 (2005), 149–180.

[14] Adrian Petcu and Boi Faltings. 2005. DPOP: A scalable method for multiagent constraint optimization. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*. 266–271.

[15] A. Petcu and B. Faltings. 2007. MB-DPOP: A New Memory-Bounded Algorithm for Distributed Optimization. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 1452–1457.

[16] Pierre Rust, Gauthier Picard, and Fano Ramparany. 2016. Using Message-Passing DCOP Algorithms to Solve Energy-Efficient Smart Environment Configuration Problems.. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*. 468–474.

[17] Ruben Stranders, Alessandro Farinelli, Alex Rogers, and Nick R Jennings. 2009. Decentralised coordination of continuously valued control parameters using the max-sum algorithm. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 601–608.

[18] Cornelis Jan van Leeuwen and Przemyslaw Pawelczak. 2017. CoCoA: A non-iterative approach to a local search (A) DCOP solver. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

[19] Thomas Voice, Ruben Stranders, Alex Rogers, and Nicholas R Jennings. 2010. A Hybrid Continuous Max-Sum Algorithm for Decentralised Coordination.. In *Proceedings of The 19th European Conference on Artificial Intelligence*. 61–66.

[20] Harel Yedidsion and Roie Zivan. 2016. Applying dcop_mst to a team of mobile robots with directional sensing abilities. In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 1357–1358.

[21] W. Yeoh, A. Felner, and S. Koenig. 2010. BnB-ADOPT: An asynchronous branch-and-bound DCOP algorithm. *Journal of Artificial Intelligence Research* 38 (2010), 85–133.