Branch-and-Bound Heuristics for Incomplete DCOPs[*]

Extended Abstract

Atena M. Tabakhi
Washington University in St. Louis
amtabakhi@wustl.edu

Yuanming Xiao
Washington University in St. Louis
yuanming.xiao@wustl.edu

William Yeoh
Washington University in St. Louis
wyeoh@wustl.edu

Roie Zivan
Ben-Gurion University of the Negev
zivanr@bgu.ac.il

ABSTRACT

The *Incomplete Distributed Constraint Optimization Problem* (I-DCOP) extends the distributed constraint optimization problem, where constraint costs are allowed to be unspecified. A distributed variant of the *Synchronous Branch-and-Bound* (SyncBB) search algorithm has been proposed to solve I-DCOPs, where unspecified constraint costs are elicited during its execution. In this paper, we propose two heuristics that can be used in conjunction with SyncBB to solve I-DCOPs. Our proposed heuristics speed up the algorithm by pruning those parts of the search space whose solution quality is sub-optimal. Thus, our model and heuristics extend the state of the art in distributed constraint reasoning to better model and solve distributed agent-based applications with user preferences.

KEYWORDS

Multi-Agent Problems; DCOPs; Heuristics; Preference Elicitation

ACM Reference Format:
Atena M. Tabakhi, Yuanming Xiao, William Yeoh, and Roie Zivan. 2021. Branch-and-Bound Heuristics for Incomplete DCOPs: Extended Abstract. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

The *Distributed Constraint Optimization Problem* (DCOP) [2, 7, 10, 13] are well-suited to model many problems that are distributed by nature and where agents need to coordinate their value assignments to minimize the aggregated constraint costs. DCOP researchers have proposed a wide variety of solution approaches, from complete search- and inference-based approaches [4, 7, 10, 17] to incomplete search-, inference-, and sampling-based methods [1, 6, 8, 9, 15, 18].

A key assumption in all of these approaches is that the constraint costs in a DCOP are known a priori. This assumption restrains the applicability of DCOPs as it does not hold in many real-world problems, where constraints encode human user preferences. As such some of these constraint costs might be unspecified and must be elicited from human users. To address this limitation, researchers proposed the *Incomplete DCOP* (I-DCOP) model [16], which *integrates* both the elicitation and solving problems into a single integrated optimization problem. In an I-DCOP, some constraint costs are unknown and can be elicited. Elicitation of unknown constraint costs will incur elicitation costs, and the goal is tofi nd a solution that minimizes the sum of constraint and elicitation costs incurred. To solve this problem, they adapted a complete algorithm – *Synchronous Branch-and-Bound* (SyncBB) [5]. In this paper, we propose heuristics that can be used by SyncBB to trade offsolution quality for faster runtimes and fewer elicitations. The proposed heuristics provides quality guarantees for I-DCOPs when elicitation costs are zero.

[*]The research was partially supported by NSF grants 1812619 and 1838364.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

2 BACKGROUND: INCOMPLETE DCOPs

An *Incomplete DCOP* (I-DCOP) [16] extends a DCOP by allowing some constraints to be partially specified. It is defined by a tuple $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{F}, \tilde{\mathcal{F}}, \mathcal{E}, \alpha \rangle$: Where $\mathcal{A} = \{a_i\}_{i=1}^{p}$ is a set of *agents*; $\mathcal{X} = \{x_i\}_{i=1}^{n}$ is a set of decision *variables*; $\mathcal{D} = \{D_x\}_{x \in \mathcal{X}}$ is a set offinite *domains*; $\mathcal{F} = \{f_i\}_{i=1}^{m}$ is a set of *constraints*, each defined over a set of decision variables: $f_i : \prod_{x \in \mathbf{x}^{f_i}} D_x \to \mathbb{R} \cup \{\infty\}$. Unlike standard DCOPs, the set of constraints $\mathcal{F}$ are not known to an I-DCOP algorithm. Instead, only the set of partially-specified constraints $\tilde{\mathcal{F}}$ are known; $\tilde{\mathcal{F}} = \{\tilde{f}_i\}_{i=1}^{m}$ is a set of *partially-specified constraints*, each defined over a set of decision variables: $\tilde{f}_i : \prod_{x \in \mathbf{x}^{f_i}} D_x \to \mathbb{R} \cup \{\infty ?\}$, where ? is a special element denoting that the cost for a given combination of value assignment is not specified. $\mathcal{E} = \{e_i\}_{i=1}^{m}$ is a set of *elicitation costs*, where each elicitation cost $e_i : \prod_{x \in \mathbf{x}^{f_i}} D_x \to \mathbb{R}$ specifies the cost of eliciting the constraint cost of a particular ? in $\tilde{f}_i$; $\alpha : \mathcal{X} \to \mathcal{A}$ is a *mapping function* that associates each decision variable to one agent. An *explored solution space* $\tilde{\mathbf{x}}$ is the union of all solutions explored so far by a particular algorithm. The *cumulative elicitation cost* $\mathcal{E}(\tilde{\mathbf{x}}) = \sum_{e \in \mathcal{E}} e(\tilde{\mathbf{x}})$ is the sum of the costs of all elicitations conducted while exploring $\tilde{\mathbf{x}}$. The *total cost* $\mathcal{F}(\mathbf{x}, \tilde{\mathbf{x}}) = \alpha_f \cdot \mathcal{F}(\mathbf{x}) + \alpha_e \cdot \mathcal{E}(\tilde{\mathbf{x}})$ is the weighted sum of both the cumulative constraint cost $\mathcal{F}(\mathbf{x})$ of solution $\mathbf{x}$ and the cumulative elicitation cost $\mathcal{E}(\tilde{\mathbf{x}})$ of the explored solution space $\tilde{\mathbf{x}}$ such that $\alpha_f + \alpha_e = 1$. The goal is tofi nd an optimal complete solution $\mathbf{x}^*$ while eliciting only a cost-minimal set of preferences from a solution space $\tilde{\mathbf{x}}^*$. More formally, the goal is to find $(\mathbf{x}^*, \tilde{\mathbf{x}}^*) = \text{argmin}_{(\mathbf{x}, \tilde{\mathbf{x}})} \mathcal{F}(\mathbf{x}, \tilde{\mathbf{x}})$.

The SyncBB [16] algorithm evaluates a node n after exploring search space $\tilde{\mathbf{x}}$, it considers the cumulative elicitation cost so far $\mathcal{E}(\tilde{\mathbf{x}})$ and the constraint costs of the CPA at node n, which we will refer to as g-values, denoted by $g(n)$. And $f(n, \tilde{\mathbf{x}}) = \alpha_f \cdot g(n) + \alpha_e \cdot \mathcal{E}(\tilde{\mathbf{x}})$ denotes the weighted sum of these values. To speed up SyncBB, one can use *cost-estimate heuristics* $h(n)$ to estimate the sum of the constraint and elicitation costs needed to complete the CPA at a particular node n. And if those heuristics are underestimates of the true cost, then they can be used to better prune the search space, that is, when $f(n, \tilde{\mathbf{x}}) = \alpha_f \cdot g(n) + h(n) + \alpha_e \cdot \mathcal{E}(\tilde{\mathbf{x}}) \geq \mathcal{F}(\mathbf{x}, \tilde{\mathbf{x}})$, where $\mathbf{x}$ is the best complete solution found so far and $\tilde{\mathbf{x}}$ is the current explored solution space.

3 BRANCH-AND-BOUND HEURISTICS

We now describe below cost-estimate heuristics that can be used in conjunction with SyncBB to solve I-DCOPs. These heuristics make use of an estimated lower bound $\mathcal{L}$ on the cost of all constraints $f \in \mathcal{F}$. Such a lower bound can usually be estimated through domain expertise and can be set to 0 in the worst case since all costs are non-negative. The more informed the lower bound, the more effective the heuristics will be in pruning the search space. Additionally, these heuristics are parameterized by two parameters – a relative weight $w \geq 1$ and an additive weight $\epsilon \geq 0$. When using these parameters, SyncBB will prune a node n if $w \cdot f(n, \tilde{\mathbf{x}}) + \epsilon \geq \mathcal{F}(\mathbf{x}, \tilde{\mathbf{x}})$: where $\mathbf{x}$ is the best complete solution found so far and $\tilde{\mathbf{x}}$ is the current explored solution space.

Child's Ancestors' Constraints (CAC) Heuristic: This heuristic is defined recursively from the leaf of the pseudo-chain (i.e., last agent in the variable ordering) used by SyncBB up to the root of the pseudo-chain (i.e.,fi rst agent in the ordering). Agent x_i in the ordering computes a heuristic value $h(x_i = d_i)$ for each of its values $d_i \in D_i$ as follows: $h(x_i = d_i) = 0$ if x_i is the leaf of the pseudo-chain. Otherwise, $h(x_i = d_i)$ is:

$$\min_{d_c \in D_c}\left[\alpha_f \hat{f}(x_i\!=\!d_i, x_c\!=\!d_c) + \alpha_e e(x_i\!=\!d_i, x_c\!=\!d_c) + h(x_c\!=\!d_c)\right] + \sum_{x_k \in A(x_c)\setminus\{x_i\}} \min_{d_k \in D_k}\left[\alpha_f \hat{f}(x_c\!=\!d_c, x_k\!=\!d_k) + \alpha_e e(x_c\!=\!d_c, x_k\!=\!d_k)\right] \quad (1)$$

where x_c is the next agent in the ordering (i.e., child of x_i in the pseudo-chain), $A(x_c)$ is the set of variables higher up in the ordering that x_c is constrained with, and each estimated cost function $\hat{f}$ corresponds exactly to a partially-specified function $\tilde{f}$, except that all the unknown costs ? are replaced with the lower bound $\mathcal{L}$. Therefore, the estimated cost $\hat{f}(\mathbf{x})$ is guaranteed to be no larger than the true cost $f(\mathbf{x})$ for any solution $\mathbf{x}$. As the heuristic of a child agent is included in the heuristic of the parent agent, these summations of costs are recursively aggregated up the pseudo-chain. It is fairly straightforward to see that this heuristic can be computed in a distributed manner – the leaf agent x_l initializes its heuristic values $h(x_l = d_l) = 0$ for all its values $d_l \in D_l$ and computes the latter term in Equation (1):

$$\sum_{x_k \in A(x_l)} \min_{d_k \in D_k}\left[\alpha_f \hat{f}(x_l\!=\!d_l, x_k\!=\!d_k) + \alpha_f e(x_l\!=\!d_l, x_k\!=\!d_k)\right] \quad (2)$$

for each of its values $d_l \in D_l$. It then sends these heuristic values and costs to its parent. Upon receiving this message, the parent agent x_p uses the information in the message to compute its own heuristic values $h(x_p\!=\!d_p)$ using Equation (1), computes the latter term similar to Equation (2) above, and sends these heuristic values and costs to its parent.

Agent's Descendants' Constraints (ADC) Heuristic: Just like the CAC heuristic, this heuristic is also defined recursively from

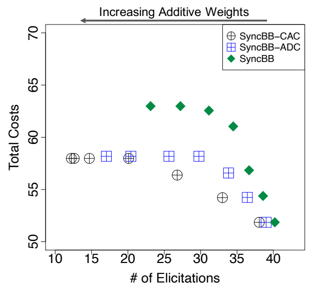

Figure 1: Varying Additive Weights, with $|\mathcal{A}| = 10$, $|\mathcal{D}| = 2$

the leaf of the pseudo-chain used by SyncBB up to the root of the pseudo-chain. Agent x_i in the ordering computes a heuristic value $h(x_i\!=\!d_i)$ for each of its values $d_i \in D_i$ as follows: $h(x_i = d_i) = 0$ if x_i is the leaf of the pseudo-chain. Otherwise, $h(x_i\!=\!d_i) =$ is:

$$\min_{d_c \in D_c}\left[\alpha_f \hat{f}(x_i\!=\!d_i, x_c\!=\!d_c) + \alpha_e e(x_i\!=\!d_i, x_c\!=\!d_c) + h(x_c\!=\!d_c)\right] + \sum_{x_j \in D(x_i)\setminus\{x_c\}} \min_{d_j \in D_j}\left[\alpha_f \hat{f}(x_i\!=\!d_i, x_j\!=\!d_j) + \alpha_e e(x_i\!=\!d_i, x_j\!=\!d_j)\right] \quad (3)$$

where x_c is the next agent in the ordering, $D(x_i)$ is the set of variables lower down in the ordering that x_i is constrained with, and each estimated cost function $\hat{f}$ is as defined for the CAC heuristic above. Both heuristics start from the leaf agent of the pseudo-chain and continue until the root agent computes its own heuristic values, at which point it starts the SyncBB algorithm.

Empirical Results: Figure 1 plots our empirical results on random graphs, where we vary the user-defined additive bound (weight) for the problems when all elicitation costs are zero. Each data point in thefi gure shows the result for one of the algorithms with one of the values of ϵ. Data points for smaller values of ϵ are in the bottom right of thefi gures and data points for larger values are in the top left of thefi gures. We plot the tradeoffs between total cost (= cumulative constraint and elicitation costs) and number of elicited costs. As expected, as the additive bound ϵ increases, the number of elicitations decreases. However, this comes at the cost of larger total costs. Between the three algorithms, SyncBB with CAC is the best.

4 CONCLUSIONS AND RELATED WORK

Incomplete DCOPs (I-DCOPs) extend DCOPs by allowing some constraints to be partially specified and the elicitation of unknown costs in such constraints incur elicitation costs. We employed the SyncBB search algorithm as the underlying solver for I-DCOPs and proposed two heuristics to speed up the algorithm by pruning the sub-optimal solutions from the search space.

Aside from I-DCOPs, researchers have also proposed the *preference elicitation problem for DCOPs* [12], where constraint costs are initially unknown and must be elicited from human users. However, they assume that the cost of eliciting constraints is uniform across all constraints which is an unrealistic assumption. In contrast, we associate non-uniform costs for elicitation of unknown constraints. Additionally, another model that is closely related is *Incomplete Weighted CSPs* [3, 11, 14], which can be seen as centralized versions of I-DCOPs.

REFERENCES

[1] Alessandro Farinelli, Alex Rogers, Adrian Petcu, and Nicholas Jennings. 2008. Decentralised Coordination of Low-Power Embedded Devices Using the Max-Sum Algorithm. In *AAMAS*. 639–646.

[2] 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.

[3] Mirco Gelain, Maria Silvia Pini, Francesca Rossi, K Brent Venable, and Toby Walsh. 2010. Elicitation strategies for soft constraint problems with missing preferences: Properties, algorithms and experimental studies. *Artificial Intelligence* 174, 3-4 (2010), 270–294.

[4] Amir Gershman, Amnon Meisels, and Roie Zivan. 2009. Asynchronous Forward-Bounding for Distributed COPs. *Journal of Artificial Intelligence Research* 34 (2009), 61–88.

[5] Katsutoshi Hirayama and Makoto Yokoo. 1997. Distributed partial constraint satisfaction problem. In *CP*. 222–236.

[6] Rajiv Maheswaran, Jonathan Pearce, and Milind Tambe. 2004. Distributed Algorithms for DCOP: A Graphical Game-Based Approach. In *PDCS*. 432–439.

[7] Pragnesh 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.

[8] Duc Thien Nguyen, William Yeoh, Hoong Chuin Lau, and Roie Zivan. 2019. Distributed Gibbs: A Linear-Space Sampling-Based DCOP Algorithm. *Journal of Artificial Intelligence Research* 64 (2019), 705–748.

[9] Brammert Ottens, Christos Dimitrakakis, and Boi Faltings. 2017. DUCT: An Upper Confidence Bound Approach to Distributed Constraint Optimization Problems. *ACM Transactions on Intelligent Systems and Technology* 8, 5 (2017), 69:1–69:27.

[10] Adrian Petcu and Boi Faltings. 2005. A Scalable Method for Multiagent Constraint Optimization. In *IJCAI*. 1413–1420.

[11] Atena M. Tabakhi. 2019. Parameterized Heuristics for Incomplete Weighted CSPs. In *AAAI*. 9898–9899.

[12] Atena M. Tabakhi, Tiep Le, Ferdinando Fioretto, and William Yeoh. 2017. Preference Elicitation for DCOPs. In *CP*. 278–296.

[13] Atena M. Tabakhi, Reza Tourani, Francisco Natividad, William Yeoh, and Satyajayant Misra. 2017. Pseudo-Tree Construction Heuristics for DCOPs and Evaluations on the ns-2 Network Simulator. In *ICTAI*. 1105–1112.

[14] Atena M. Tabakhi, William Yeoh, and Makoto Yokoo. 2019. Parameterized Heuristics for Incomplete Weighted CSPs with Elicitation Costs. In *AAMAS*. 476–484.

[15] Meritxell Vinyals, Juan Rodríguez-Aguilar, and Jesús Cerquides. 2011. Constructing a Unifying Theory of Dynamic Programming DCOP Algorithms via the Generalized Distributive Law. *Journal of Autonomous Agents and Multi-Agent Systems* 22, 3 (2011), 439–464.

[16] Yuanming Xiao, Atena M. Tabakhi, and William Yeoh. 2020. Embedding Preference Elicitation Within the Search for DCOP Solutions. In *AAMAS*. 2044–2046.

[17] William Yeoh, Ariel Felner, and Sven Koenig. 2010. BnB-ADOPT: An Asynchronous Branch-and-Bound DCOP Algorithm. *Journal of Artificial Intelligence Research* 38 (2010), 85–133.

[18] Weixiong Zhang, Guandong Wang, Zhao Xing, and Lars Wittenburg. 2005. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence* 161, 1-2 (2005), 55–87.