Preserving Privacy in Region Optimal DCOP Algorithms

Tamir Tassa
The Open University
Ra'anana, Israel
tamirta@openu.ac.il

Roie Zivan
Ben-Gurion University of the Negev
Beer-Sheva, Israel
zivanr@bgu.ac.il

Tal Grinshpoun
Ariel University
Ariel, Israel
talgr@ariel.ac.il

Abstract

Region-optimal algorithms are local search algorithms for the solution of Distributed Constraint Optimization Problems (DCOPs). In each iteration of the search in such algorithms, every agent selects a group of agents that comply with some selection criteria (each algorithm specifies different criteria). Then, the agent who selected the group, called the mediator, collects assignment information from the group and neighboring agents outside the group, in order to find an optimal set of assignments for its group's agents. A contest between mediators of adjacent groups determines which groups will replace their assignments in that iteration to the found optimal ones. In this work we present a framework called RODA (Region-Optimal DCOP Algorithm) that encompasses the algorithms in the region-optimality family, and in particular any method for selecting groups. We devise a secure implementation of RODA, called P-RODA, which preserves constraint privacy and partial decision privacy. The two main cryptographic means that enable this privacy preservation are secret sharing and homomorphic encryption. We estimate the computational overhead of P-RODA with respect to RODA and give an upper bound that depends on the group and domain sizes and the graph topology but not on the number of agents. The estimations are backed with experimental results.

1 Introduction

The Distributed Constraint Optimization Problem (DCOP) is a general model for solving distributed combinatorial problems that has a wide range of applications in multi-agent systems. Complete algorithms for DCOP-solving [Gershman *et al.*, 2009; Modi *et al.*, 2005; Petcu and Faltings, 2005] are guaranteed to find the optimal solution, but because DCOPs are NP-hard, these algorithms' worst case runtime is exponential. Thus, there is growing interest in incomplete algorithms, which may find sub-optimal solutions but run quickly enough to be applied to large problems or real-time applications [Maheswaran *et al.*, 2004; Teacy *et al.*, 2008; Zhang *et al.*, 2005; Zivan *et al.*, 2014].

Local search algorithms, e.g., DSA [Zhang *et al.*, 2005] and DBA [Hirayama and Yokoo, 2005], are, in general, simple incomplete algorithms that were found empirically to produce high-quality solutions. The main disadvantage of these algorithms is that they do not offer guarantees on the quality of the solutions that they produce. Thus, in the last decade, researchers have developed solution concepts that offer a balance in the trade-off between run-time efficiency (the time required to find a solution) and the guaranteed solution quality [Katagishi and Pearce, 2007; Kiekintveld *et al.*, 2010; Maheswaran *et al.*, 2006; Pearce and Tambe, 2007; Vinyals *et al.*, 2011]. The guarantees are achieved by selecting a criterion according to which agents are grouped, and then producing solutions that are locally optimal in the following sense: there exists no better-quality solution that differs from the obtained solution only in the assignments of agents that are all contained in a single group. The criteria for selecting groups are defined by two parameters: k, an upper bound on the group's size, and t, an upper bound on the distance (i.e., the length of the shortest connecting path in the constraint graph) of agents in the group from the group's central agent, called the *mediator*. These two parameters define for each agent, serving as a mediator, a *region*, which is the collection of all maximal-sized connected groups of at most k agents that include the mediator and in which all agents are of distance at most t from the mediator [Vinyals *et al.*, 2011]. Such solutions, termed *region-optimal*, provide under certain conditions a guaranteed bound on the ratio between their quality and the quality of an optimal solution [Pearce and Tambe, 2007; Vinyals *et al.*, 2011].

Privacy is one of the main motivations for solving constraint problems in a distributed manner. The term privacy is quite broad, a fact that gave rise to several categorizations of the different types of privacy [Faltings *et al.*, 2008; Greenstadt *et al.*, 2007; Grinshpoun, 2012]. In this paper we relate to the categorization of Faltings et al. [2008] that distinguishes between agent privacy, topology privacy, constraint privacy, and decision privacy. (For full definitions of those notions the reader is referred to [Faltings *et al.*, 2008] or [Léauté and Faltings, 2013]; we briefly recall those definitions in Section 4.)

The first attempt to produce a more secure region-optimal algorithm (actually k-optimal) was conducted by Greenstadt [2009]. While the ideas presented in that extended ab-

stract were preliminary, we share the motivation and take herein a further step in that direction.

We describe here an algorithmic framework called RODA (Region-Optimal DCOP Algorithm) that generalizes, for didactic purposes, existing local search algorithms that issue solutions with quality guarantees, e.g. the KOPT algorithm [Katagishi and Pearce, 2007], or the DALO algorithm [Kiekintveld *et al.*, 2010]. RODA is a new formalization rather than a new algorithm; it is an umbrella setup that generalizes the main existing region-optimal algorithms, and allows us to include in our study of privacy the region selection methods of all existing algorithms of the region-optimality family [Katagishi and Pearce, 2007; Kiekintveld *et al.*, 2010; Vinyals *et al.*, 2011].

We then proceed to present P-RODA, a privacy-preserving implementation of RODA. Hence, P-RODA includes a privacy-preserving implementation of a general region-optimal algorithm, which can implement the region traversing methods of KOPT, DALO, and any other algorithm from the region-optimality family, and follow either a synchronous or an asynchronous operation mode. P-RODA is a perfect simulation of RODA, in the sense that given the same random choices (some of the algorithms that fall under the RODA framework use randomness), both RODA and P-RODA will go through the same sequence of intermediate assignments and will issue the same output after a given number of iterations. However, centralized computations in RODA that may leak private information are replaced in P-RODA with distributed computations that prevent such information leakage. Thus, our framework securely achieves k-size-optimality, t-distance-optimality, or any combination of the two, and finds solutions with the same guarantees on the distance from the optimum as proved in [Kiekintveld *et al.*, 2010; Pearce and Tambe, 2007; Vinyals *et al.*, 2011].

Terminology and notations. A Distributed Constraint Optimization Problem (DCOP) [Hirayama and Yokoo, 1997] is a tuple $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a set of agents $A_1, A_2, \ldots, A_n$, $\mathcal{X}$ is a set of variables $X_1, X_2, \ldots, X_n$[1], $\mathcal{D}$ is a set of finite domains $D_1, D_2, \ldots, D_n$, and $\mathcal{R}$ is a set of binary relations (constraints). Each variable X_i takes values in the domain D_i, and it is held by a single agent. A constraint $C_{i,j} \in \mathcal{R}$ defines a non-negative cost for every possible value combination of X_i and X_j, for some $1 \leq i < j \leq n$. A *value assignment* is a pair including a variable and a value from that variable's domain. A *complete assignment* consists of value assignments to all variables in $\mathcal{X}$. The objective is to find a complete assignment of minimal cost. The *constraint graph* is a graph whose nodes are the n agents (or variables) where two nodes are connected by an edge if the two corresponding variables are constrained. We shall refer to such pairs of agents or variables as *neighbors*. For every agent A_i, $1 \leq i \leq n$, we let $N_t(A_i)$ denote the set of all agents whose distance to A_i in the constraint graph is at most t. The special case of the distance-1 neighborhood will be denoted $N(A_i) := N_1(A_i)$.

[1]We make a standard assumption that each agent holds exactly one variable.

2 Region Optimal DCOP Algorithms

Region-optimality is a concept that is based on assigning agents to groups, such that each group of agents has one agent that performs as its mediator. For a given agent A_h, its region R_h is the collection of all groups of which A_h can serve as the mediator. A region is commonly defined by two parameters, k and t:

Definition 1 *The region $R_h = R_h^{(k,t)}$ of agent A_h is the collection of all subsets of agents $B \subset \mathcal{A}$ such that: (a) $A_h \in B$, (b) $|B| \leq k$, (c) $B \subseteq N_t(A_h)$, (d) the restriction of the constraint graph to B is connected, and (e) there exists no proper superset of B that complies with all previous conditions.*

Definition 2 *Given a DCOP with n agents and the entire set of corresponding regions $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$, a region-optimal solution to this DCOP is a complete assignment whose cost cannot be reduced by changing the value assignments only to agents that are all included in a single group that belongs to some region in $\mathcal{R}$.*

Protocol 1 describes RODA, an algorithmic framework that generalizes existing region-optimal algorithms [Katagishi and Pearce, 2007; Kiekintveld *et al.*, 2010; Vinyals *et al.*, 2011]. We proceed to explain it in detail. Hereinafter, when we speak of a general agent we denote it by A_i; however, when we speak of that agent as a mediator we denote it by A_h, $1 \leq i, h \leq n$.

RODA starts with an initialization phase (Step 1), in which each agent A_h gathers information from agents in its t-distance neighborhood $N_t(A_h)$; the collected information is needed for A_h to determine its region R_h (Definition 1), and later on to function as the mediator. That information includes: (a) the domains of variables controlled by agents in that neighborhood; and (b) the constraints that agents in that neighborhood have with all their neighbors (these constraints can involve agents of distance at most $t+1$ from A_h).

After the initialization phase ends, the local search phase starts. First, each agent selects an initial assignment to its variable (Step 2). Then, the algorithm performs a fixed number L of improvement iterations. In the ℓth iteration (Steps 3-10), some of the agents will update their current assignment from $a_i^{\ell-1}$ to a_i^{ℓ} towards reducing the overall cost. After all agents initiate the next assignment to be like the current one (Step 4), every agent A_h selects a group $\mathcal{A}_h^{\ell}$ from its region R_h (Step 5); the selection can be random or one that follows some systematic rule. In the next steps A_h will act as the mediator of the selected group.

Then (Step 6), each mediator A_h gathers the current assignments of all agents in $\mathcal{A}_h^{\ell}$ as well as of their direct neighbors (some of whom could be outside $\mathcal{A}_h^{\ell}$). With that information, A_h finds the locally best assignments β for the agents in its group $\mathcal{A}_h^{\ell}$ (Step 7). Then, A_h computes the improvement Δ_h^{ℓ} in the overall cost, in case the current assignments in its group are replaced with the found locally best ones (Step 8). It broadcasts its findings (β and Δ_h^{ℓ}) to its group (Step 9). Later on we explain in detail how A_h computes β for its group and the corresponding Δ_h^{ℓ}.

Next, a contest takes place between neighboring groups (groups that include agents that are neighbors). Every group

which is a local winner (namely, the improvement which it offers is greater than the improvement offered by any of its neighboring groups, where ties are broken by the groups' indices) updates its current assignments to the found local optimal ones β (Step 10). After conducting the pre-set number of iterations L, the agents set their variables to the last assignment found (Step 11).

Note that our description of RODA (and P-RODA) adopts the synchronous operation of KOPT [Katagishi and Pearce, 2007]. The adaptation to an asynchronous operation as in DALO [Kiekintveld *et al.*, 2010] is simple and omitted due to space limitation.

We now revisit Steps 7 and 8 and provide the necessary technical details. Let

• $D_h^\ell := \prod_{A_i \in \mathcal{A}_h^\ell} D_i$ be the Cartesian product of the domains of all variables of $\mathcal{A}_h^\ell$, and $N_h^\ell := \prod_{A_j \in (\bigcup_{A_i \in \mathcal{A}_h^\ell} N(A_i)) \setminus \mathcal{A}_h^\ell} D_j$ be the Cartesian product of the domains of all variables that correspond to agents outside $\mathcal{A}_h^\ell$ that have neighbors in $\mathcal{A}_h^\ell$.

• $\beta_h^{\ell-1} \in D_h^\ell$ be the current partial assignment to the variables controlled by $\mathcal{A}_h^\ell$, and $\alpha_h^{\ell-1} \in N_h^\ell$ be the partial assignment that was received by the agents in $\mathcal{A}_h^\ell$ from their neighbors outside $\mathcal{A}_h^\ell$. ($\beta_h^{\ell-1}$ and $\alpha_h^{\ell-1}$ consist of the assignments that A_h retrieves in Step 6.)

Then, A_h finds a $\beta \in D_h^\ell$ that minimizes the local cost

$$\hat{C}(\beta, \alpha_h^{\ell-1}) := \sum_{A_i \in \mathcal{A}_h^\ell, A_j \in N(A_i) \cap \mathcal{A}_h^\ell, j > i} C_{i,j}(\beta|_{D_i}, \beta|_{D_j}) + \sum_{A_i \in \mathcal{A}_h^\ell, A_j \in N(A_i) \setminus \mathcal{A}_h^\ell} C_{i,j}(\beta|_{D_i}, \alpha_h^{\ell-1}|_{D_j}) ; \quad (1)$$

Here, the notation $\beta|_{D_i}$ denotes the D_i-entry of the tuple β. The first term on the right-hand side of Eq. (1) is the sum of constraints that involve two agents inside the group $\mathcal{A}_h^\ell$. The second term is the sum of constraints that involve one agent in $\mathcal{A}_h^\ell$ and one external agent. Finally, given the found optimal tuple β, A_h computes

$$\Delta_h^\ell := \hat{C}(\beta_h^{\ell-1}, \alpha_h^{\ell-1}) - \hat{C}(\beta, \alpha_h^{\ell-1}) \geq 0 , \quad (2)$$

which is the improvement in the local cost for $\mathcal{A}_h^\ell$ if they replace the current local partial assignment $\beta_h^{\ell-1}$ with β.

3 Private RODA

In this section we explain how to execute the RODA algorithm in a privacy-preserving manner. To that end, we follow the algorithmic flow of Protocol 1 and explain, for each step in that protocol, how to execute it in a privacy-preserving manner. We make the standard assumption (e.g. [Grinshpoun and Tassa, 2014; Léauté and Faltings, 2013; Tassa *et al.*, 2015]) that the agents are semi-honest, which means that they respect the protocol and do not form coalitions.

The main effort in transforming RODA into a private algorithm (to which we refer hereinafter by P-RODA) lies in converting the centralized computations that are carried out by the mediator (especially in Steps 7 and 8) to distributed computations that are performed by all agents in the group using secure multi-party protocols. In RODA, in order to allow the mediator to perform the centralized computations for its group, it had to receive in the initialization phase (Step 1) and then in each iteration (Step 6) private information from agents that are at distance at most $t+1$ from it. As such exchange of data contradicts privacy, we replace it in P-RODA with a much reduced exchange of data and then we modify the subsequent steps so that they are performed by all agents in the group using secure multi-party protocols. In what follows, we provide explanations only on those steps in Protocol 1 that have to be modified in P-RODA; steps that we do not mention below remain the same.

Protocol 1 RODA

1: The agents exchange local information (see details in the text).
2: A_i randomly selects $a_i^0 \in D_i$, $1 \leq i \leq n$.
3: **for** $\ell = 1, \ldots, L$ **do**
4: A_i sets $a_i^\ell = a_i^{\ell-1}$, $1 \leq i \leq n$.
5: A_h selects $\mathcal{A}_h^\ell \in R_h$, $1 \leq h \leq n$.
6: A_h receives $a_j^{\ell-1}$ from A_j for all $A_j \in \bigcup_{A_i \in \mathcal{A}_h^\ell} N(A_i)$.
7: A_h finds a locally optimal tuple of assignments, $\beta \in D_h^\ell := \prod_{A_i \in \mathcal{A}_h^\ell} D_i$, for the variables controlled by its group $\mathcal{A}_h^\ell$.
8: A_h computes Δ_h^ℓ, the cost improvement if all agents in $\mathcal{A}_h^\ell$ update their assignment to the one given by β.
9: A_h informs all agents in $\mathcal{A}_h^\ell$ about the found β and Δ_h^ℓ.
10: For every $1 \leq h \leq n$: If $\mathcal{A}_h^\ell$ wins the contest against $\mathcal{A}_{h'}^\ell$ for all groups $\mathcal{A}_{h'}^\ell$ that are neighboring to $\mathcal{A}_h^\ell$, then $\forall A_i \in \mathcal{A}_h^\ell$, A_i sets a_i^ℓ according to β. (Winning occurs if $\Delta_h^\ell > \Delta_{h'}^\ell$ or if $\Delta_h^\ell = \Delta_{h'}^\ell$ and $h < h'$.)
11: A_i sets $X_i := a_i^L$, $1 \leq i \leq n$.

• **Step 1 (Initialization).** This phase is performed in P-RODA in a reduced manner. Every agent A_h, $1 \leq h \leq n$, learns only topological information in the form of the restriction of the constraint graph to $N_t(A_h)$. This is sufficient for A_h to determine its region R_h. All the other information that A_h collected at this phase in RODA from its neighbors – the domains of variables for all agents in $N_t(A_h)$, and the constraint information relating to each pair of agents of which one agent is from $N_t(A_h)$ – is not needed in P-RODA, since in P-RODA we replace the centralized computations that A_h did as a mediator in RODA with secure multi-party protocols.

• **Step 6 (Assignment propagation).** While in RODA every mediator learns the current assignment data for all agents in its group and their neighbors (which means that it may learn assignment data of agents which are at distance as far as $t+1$ from it), P-RODA performs a much reduced propagation of current assignment data: every agent A_i sends its current assignment $a_i^{\ell-1}$ only to its direct neighbors ($N(A_i) \setminus \{A_i\}$).

• **Step 7 (Locally optimal assignments).** This step includes the computations that are hardest to perform securely, whence we postpone the discussion of their private execution to Section 3.1. The sub-protocol described in Section 3.1 ends with all agents in the group learning the currently optimal β; hence, that protocol accomplishes Step 7 and part of Step 9 in RODA. In P-RODA, as opposed to RODA, it is essential to have all agents in the group know β before proceeding to

computing Δ_h^ℓ in Step 8, because that computation is done in P-RODA by a distributed protocol involving all agents.

• **Steps 8-9 (Local improvements).** While in RODA it was the mediator who computed Δ_h^ℓ, and then informed all agents in its group about the result, in P-RODA all agents in the group compute it jointly and privately using a secure summation protocol. Consider the local cost $\hat{C}(\beta, \alpha_h^{\ell-1})$, as defined in Eq. (1); it can be broken into the following sum,

$$\hat{C}(\beta, \alpha_h^{\ell-1}) = \sum_{A_i \in \mathcal{A}_h^\ell} \gamma_i(\beta, \alpha_h^{\ell-1}), \quad (3)$$

where

$$\gamma_i(\beta, \alpha_h^{\ell-1}) := \sum_{A_j \in N(A_i) \cap \mathcal{A}_h^\ell, j>i} C_{i,j}(\beta|_{D_i}, \beta|_{D_j}) + \sum_{A_j \in N(A_i) \setminus \mathcal{A}_h^\ell} C_{i,j}(\beta|_{D_i}, \alpha_h^{\ell-1}|_{D_j}). \quad (4)$$

It is easy to see that $\Delta_h^\ell = \sum_{A_i \in \mathcal{A}_h^\ell} \Delta_h^\ell(i)$, where $\Delta_h^\ell(i) := \gamma_i(\beta_h^{\ell-1}, \alpha_h^{\ell-1}) - \gamma_i(\beta, \alpha_h^{\ell-1})$ is a value known to $A_i \in \mathcal{A}_h^\ell$ (because A_i knows β from Step 7 and it also knows the relevant components of $\beta_h^{\ell-1}$ and $\alpha_h^{\ell-1}$ from the reduced Step 6 in P-RODA). Therefore, Δ_h^ℓ can be computed by all agents in $\mathcal{A}_h^\ell$ using a secure summation protocol.

However, secure summation protocols usually reveal the final sum to at least one of the agents, what might hinder constraint privacy. Therefore, we perform the secure summation protocol with a small "twist" so that instead of issuing the final sum Δ_h^ℓ, it issues two shares, denoted s_h and s'_h, that distribute uniformly at random on $\mathbb{Z}_S$ for some large integer S, and $\Delta_h^\ell = s_h + s'_h \mod S$. The mediator will get s_h while all other members of its group will get s'_h. Using such secret sharing implies that, under the semi-honestness assumption, none of the agents learns *any* information on Δ_h^ℓ.

• **Step 10 (Contest).** While in RODA, the difference Δ_h^ℓ was known to the mediator, whence it was easy for mediators of neighboring groups to compare those values in order to determine which of the groups are winners in this iteration (and, consequently, get to update their local assignments), in P-RODA the value of Δ_h^ℓ is shared by the mediator and any other single member of its group. Therefore, we proceed to describe the manner in which the contest of Step 10 can be carried out in P-RODA.

First, the agents in each group $\mathcal{A}_h^\ell$ engage in a secure union protocol [Tassa and Gudes, 2012] for finding the set of all neighboring groups; we omit the details of that procedure. Then, each pair of neighboring groups, say $\mathcal{A}_h^\ell$ and $\mathcal{A}_m^\ell$, where $h < m$, performs the following verification:

- A_h and A_m select "deputies" in their respective groups, $A'_h \in \mathcal{A}_h^\ell \setminus \{A_h\}$ and $A'_m \in \mathcal{A}_m^\ell \setminus \{A_m\}$.
- A_h sends its share in Δ_h^ℓ, s_h, to A_m.
- A_m sends $(s_h - s_m) \mod S$ to A'_h.
- A'_h sends $(s_h - s_m + s'_h) \mod S$ to A'_m.
- A'_m computes $(s_h - s_m + s'_h - s'_m) \mod S$. This value equals $(\Delta_h^\ell - \Delta_m^\ell) \mod S$.
- A'_m infers from the last difference whether $\Delta_h^\ell \geq \Delta_m^\ell$ or not. If so, it informs A_h who informs all agents in $\mathcal{A}_h^\ell$ that their group won. Otherwise, it informs all agents in $\mathcal{A}_m^\ell$ that their group won. (Notice that, as in RODA, in case of a tie, the group with the lower index wins.)

For the sake of achieving constraint privacy (see Theorem 2 later on), it is important that the agent who gets the difference $(\Delta_h^\ell - \Delta_m^\ell) \mod S$ will not be a member of both $\mathcal{A}_h^\ell$ and $\mathcal{A}_m^\ell$. Indeed, as shown by Kiekintveld *et al.* [2010], it is possible to select the deputies A'_h and A'_m so that at least one of them is not a member of both $\mathcal{A}_h^\ell$ and $\mathcal{A}_m^\ell$.

3.1 Private Computation of the Best Partial Assignment for a Group of Agents

Overview

Here we discuss the privacy-preserving execution of Step 7 in RODA. In that step, the agents within each group need to solve a local DCOP. Hence, one possibility would be to invoke an existing privacy-preserving complete algorithm for that purpose. Two recent studies proposed such algorithms: Grinshpoun and Tassa [2014] proposed P-SyncBB, a privacy-preserving version of the SyncBB algorithm; and Léauté and Faltings [2013] proposed three privacy-preserving versions of DPOP: P-DPOP$^{(+)}$, P$^{3/2}$-DPOP$^{(+)}$, and P^2-DPOP$^{(+)}$.

One may implement any of those four algorithms in the framework of P-RODA for a privacy-preserving execution of Step 7 (after a simple modification in order to take into account also constraints vis-a-vis agents outside the group). However, all these algorithms have their shortcomings: P-SyncBB incurs a large number of communication rounds and messages, and disrespects decision privacy; P-DPOP$^{(+)}$ and P$^{3/2}$-DPOP$^{(+)}$ ensure only partial constraint privacy; and P^2-DPOP$^{(+)}$ is very inefficient (in terms of run-time). Hence, we proceed to describe here a novel protocol for the secure solution of the local DCOP within a group. The protocol can be executed efficiently only when the Cartesian products of the domains of the group's variables, D_h^ℓ, are not too large. Thus, we suggest to use it whenever feasible, and when it is not to use one of the above mentioned algorithms.

A Statement of the Computational Problem

Given $\alpha_h^{\ell-1} \in N_h^\ell$, the partial assignment that was received by the agents in $\mathcal{A}_h^\ell$ from their neighbors outside $\mathcal{A}_h^\ell$ (Step 6 in P-RODA), the agents in $\mathcal{A}_h^\ell$ wish to compute a tuple $\beta \in D_h^\ell = \prod_{A_i \in \mathcal{A}_h^\ell} D_i$ that minimizes the local cost $\hat{C}(\beta, \alpha_h^{\ell-1})$, Eq. (1), in a privacy-preserving manner. (Note that $\alpha_h^{\ell-1}$ is not known in its entirety to any single agent in $\mathcal{A}_h^\ell$; each agent in the group knows only those components in $\alpha_h^{\ell-1}$ that correspond to its direct neighbors outside the group.)

Arranging the Partial Costs in a Tensor

For simplicity, let us assume that $\mathcal{A}_h^\ell = \{A_1, A_2, \ldots, A_k\}$. Assume further that all domains D_j, $1 \leq j \leq n$, are publicly ordered: $D_j = \{v_0^j, v_1^j, \ldots, v_{|D_j|-1}^j\}$. Then each $A_i \in \mathcal{A}_h^\ell$ can construct a private k-dimensional tensor G_i such that for any multi-index $[i_1, \ldots, i_k]$, where $0 \leq i_j \leq$

$|D_j| - 1, 1 \leq j \leq k$, the corresponding entry in the tensor is $G_i[i_1, i_2, \ldots, i_k] = \gamma_i(\beta, \alpha_h^{\ell-1})$, where $\beta = (v_{i_1}^1, \ldots, v_{i_k}^k)$, and γ_i is as defined in Eq. (4). According to Eq. (3), the sum of these tensors is a tensor G in which $G[i_1, i_2, \ldots, i_k] = \hat{C}(\beta, \alpha_h^{\ell-1})$, for $\beta = (v_{i_1}^1, \ldots, v_{i_k}^k)$. To summarize: each of the agents $A_i \in \mathcal{A}_h^\ell$ holds a private tensor G_i. They jointly wish to compute the multi-index of the entry in $G = \sum_{i=1}^k G_i$ which is minimal. That multi-index reveals the required β. The protocol below performs that.

The Protocol

Let $E : \mathcal{G}_P \to \mathcal{G}_C$ be a public-key encryption function from some additive group of plaintexts $\mathcal{G}_P$ to a multiplicative group of ciphertexts $\mathcal{G}_C$. Assume that: (a) its decryption key is known only to the mediator A_h; (b) it is probabilistic (i.e., encryption depends also on a random string); and (c) it is homomorphic (i.e., $\forall m, m' \in \mathcal{G}_P$, $E(m + m') = E(m) \cdot E(m')$). Paillier cryptosystem [Paillier, 1999] is an example of such an encryption function. Then, the protocol goes as follows:

1. The mediator A_h selects a deputy A'_h.
2. Agent A_i, for all $1 \leq i \leq k$, sends to A'_h the component-wise encryption of its private tensor $E(G_i)$.
3. A'_h computes the component-wise product $\prod_{i=1}^k E(G_i)$. Owing to the homomorphic property, A'_h gets as a result the encrypted tensor $E(\sum_{i=1}^k G_i) = E(G)$.
4. A'_h selects a random $c \in \mathcal{G}_C$ and computes $c \cdot E(G)$. Since $c = E(r)$ for some random $r \in \mathcal{G}_P$, it holds that $c \cdot E(G) = E(r) \cdot E(G) = E(G + r)$ (the multiplication with the scalar c and the addition with the scalar r are done component-wise).
5. A'_h selects a secret and random permutation π on $D_h^\ell = \prod_{i=1}^k D_i$ and then sends to the mediator A_h the permuted tensor $\pi(E(G + r)) = E(\pi(G + r))$.
6. A_h decrypts and recovers the tensor $\pi(G + r)$.
7. A_h informs A'_h of the position in $\pi(G + r)$ of the minimal entry.
8. A'_h uses π^{-1} to recover the original multi-index of the minimal entry. If it is $[i_1, i_2, \ldots, i_k]$ then $\beta = (v_{i_1}^1, \ldots, v_{i_k}^k)$. A'_h informs all group members about β.

A note about finding the minimal entry: We view the entries of G (and $\pi(G)$) as integers, as they describe local costs of tuples in D_h^ℓ. We wish to find a minimal entry in $\pi(G)$. However, the entries in $\pi(G + r) = \pi(G) + r$ are the result of addition modulo q (and not an addition of integers). So such an addition may create a wrap around which, usually, will prevent A_h from finding a minimal entry in $\pi(G)$. However, as q is typically a very large integer, and in particular larger than twice the maximal entry in $\pi(G)$, A_h can infer from $\pi(G) + r$ the location of a minimal entry in $\pi(G)$.

3.2 P-RODA Simulates RODA

An important observation about the relation between RODA and P-RODA is the following.

Theorem 1 *P-RODA perfectly simulates RODA in the following sense: When the two algorithms are executed on the same DCOP setting, then assuming that both start with the same random initial assignment $(a_1^0, \ldots, a_n^0)$ and that in each iteration every mediator selects the same group from its region in P-RODA as it does in RODA, then the sequence of intermediate assignments $(a_1^\ell, \ldots, a_n^\ell)$, $\ell \geq 1$, that the two algorithms produce will be the same.*

A consequence of Theorem 1 is that all convergence guarantees for RODA, as implied by the proofs in [Kiekintveld *et al.*, 2010; Pearce and Tambe, 2007; Vinyals *et al.*, 2011], equally apply to P-RODA.

4 Privacy Discussion

In this section we discuss the privacy of P-RODA with respect to the common notions of privacy in this field [Faltings *et al.*, 2008; Léauté and Faltings, 2013]. Due to space limitations, we omit proofs.

Theorem 2 *P-RODA maintains constraint privacy in the sense that no agent may use its view during P-RODA's execution in order to infer binary constraints of other agents.*

Theorem 3 *P-RODA respects partial decision privacy: it leaks to any A_i only the final decisions of agents in $N(A_i)$ that were in the same group with A_i in the last iteration; the final decisions of all other agents remain unknown to A_i.*

We conclude with a note about agent and topology privacy. Agent privacy is respected when no agent can discover the identity or even the existence of non-neighboring agents [Léauté and Faltings, 2013, Definition 4]. Topology privacy is respected when no agent is able to discover the existence of topological constructs in the constraint graph, such as nodes (i.e. variables) or edges (i.e. constraints), unless it owns a variable involved in the construct [Léauté and Faltings, 2013, Definition 5]. It seems that RODA is inconsistent with either of those two privacy notions in the sense that an algorithm that achieves one of those privacy goals will essentially differ in its operation from RODA. Indeed, for any $t > 1$, each mediator has to gather around it a group of agents in a distance up to t from it. Clearly, such a group reveals to the mediator the existence of agents that are not direct neighbors. Moreover, the group must be connected, so that even if the group is selected by a trusted third party, the mediator learns topological properties of the graph which topology privacy forbids.

5 Efficiency Analysis

We analyze here the computational overhead of P-RODA with respect to the baseline RODA. We count only encryption and decryption operations, since the other performed operations (e.g. random numbers' generation, additions, or multiplications) have computational costs that are orders of magnitude smaller than those of the cryptographic operations.

Let us fix an agent A_i, $1 \leq i \leq n$. We denote by p_i^ℓ the number of groups to which it belongs in the ℓth iteration, and by $H_i^\ell := \{h_{i,j}^\ell : 1 \leq j \leq p_i^\ell\}$ the indices of those groups. Then the excessive computational cost for A_i is to perform $\sum_{j=1}^{p_i^\ell} |D_{h_{i,j}^\ell}^\ell|$ encryptions. In addition, as A_i is the mediator

of its own group, $\mathcal{A}_i^\ell$, it needs to perform also $|D_i^\ell|$ decryptions. (Those costs are incurred by the protocol in Section 3.1.) Since an agent may be selected for groups only of mediators within distance t from it, and since every group has at most k agents, we infer the following.

Proposition 4 *The computational overhead in each iteration of P-RODA is bounded by* $d^k n_t C_E + d^k C_D$ *where* $d := \max_{1 \le i \le n} |D_i|$ *is the maximal domain size,* $n_t := \max_{1 \le i \le n} |N_t(A_i)|$ *is the maximal size of a* t*-neighborhood, and* C_E *and* C_D *are the costs of encryption and decryption, respectively. In particular, the computational overhead does not depend on the number of agents* n*.*

To realize the actual time it will take P-RODA to run we followed the *simulated time* approach [Sultanik *et al.*, 2008] by measuring the time of atomic operations performed in the algorithm and then counting the non-concurrent times these operations are performed. We measured the run-times of encryption and decryption by averaging multiple runs of the common Java implementation of the Paillier cryptosystem[2] on a hardware comprised of an Intel i7-4600U processor and 16GB memory. Our tests show that encryption takes at most $C_E = 2$ msec, while decryption takes at most $C_D = 3$ msec.

According to Proposition 4 the factors that impact the computational overhead are k, d, and the topology of the t-distance neighborhoods. As the dependence on d is polynomial while the dependence on k is exponential we focus here on the latter dependence. In order to understand the effect of the neighborhood's topology, we consider three classic types of networks – random networks (Erdős-Rényi model [1959]), scale-free networks (Barabási-Albert model [1999]), and small-world networks (Watts-Strogatz model [1998]).

Figure 1 depicts the computational overhead as a function of k in the three network types, when running the algorithm for $L = 50$ iterations. In this setup we fix the number of agents to $n = 100$, the constraint density to $p = 0.1$, the domain sizes to $d = 5$, and the upper bound of the distance to $t = 1$, and vary $k = 3, \ldots, 8$. Figure 1 confirms the exponential dependency on k, whence for high values of k one may need to switch in Step 7 to another complete DCOP private algorithm such as P-SyncBB. Another interesting phenomenon is the higher computational overhead in scale-free networks. This is not surprising, since scale-free networks consist of several highly-connected hubs, and that directly affects n_t. Nevertheless, when using higher distance values ($t \ge 3$), scale-free networks exhibit similar performance to that of the other network types. Varying the distance t has almost no effect on the performance of random and small-world networks, and therefore these results are omitted.

Although not directly affecting the computational overhead, an increase in the number of agents n might potentially affect the topology of neighborhoods, and lead to an indirect effect on the computational overhead. We therefore use the same setting of the previous experiment, but now we fix $k = 3$ and vary $n = 100, \ldots, 1000$. Figure 2 confirms that the problem's size has almost no effect on the computational overhead, except for the case of scale-free networks in which

[2]http://www.csee.umbc.edu/ kunliu1/research/Paillier.html

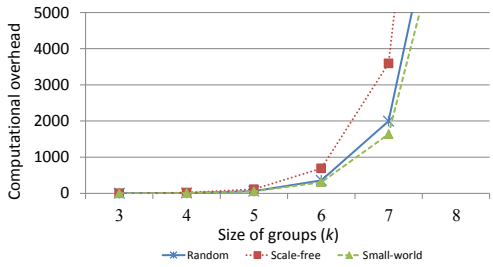

Figure 1: Computational overhead (minutes) as a function of k.

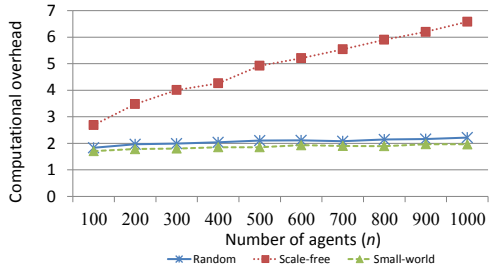

Figure 2: Computational overhead (minutes) as a function of n.

n_t does increase when n increases and, consequently, we witness a moderate increase in the computational overhead.

6 Conclusion

Region optimality is an important concept that offers a balance between run-time efficiency and guarantees on the solution quality. Privacy loss is a major drawback of region-optimal algorithms, yet, to the best of our knowledge, no secure protocol for finding region-optimal solutions has been proposed to date. In this paper we proposed P-RODA, an algorithmic framework that converges to region-optimal solutions while preserving constraint privacy and partial decision privacy. The underlying algorithm RODA generalizes the algorithms in the region-optimality family, such as KOPT [Katagishi and Pearce, 2007] and DALO [Kiekintveld *et al.*, 2010]. Privacy is achieved chiefly by using secret sharing and homomorphic encryption. P-RODA is scalable with respect to the number of agents n, but it is more suitable for regions with a small group size k.

As future work we aim at improving the privacy guarantees of P-RODA. One such improvement could be the obtaining of full decision privacy. Currently, P-RODA reveals to every agent the final decision of all direct neighboring agents that happened to be with that agent in the same group in the last iteration. It is possible to achieve *full* (rather than partial) decision privacy by enhancing the secure implementation of Step 7 in P-RODA so that each agent learns only its component in the optimal local assignment β and not all of the tuple β, as is the case now. Similarly, it might be possible to achieve assignment privacy by replacing the transfer of current assignment data in Step 6 of P-RODA with secure multi-party protocols. Clearly, such enhancements of privacy come with a price tag, as they entail greater computational and communication costs.

References

[Barabási and Albert, 1999] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[Erdős and Rényi, 1959] Paul Erdős and Alfréd Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[Faltings *et al.*, 2008] B. Faltings, T. Léauté, and A. Petcu. Privacy guarantees through distributed constraint satisfaction. In *WI-IAT*, pages 350–358, 2008.

[Gershman *et al.*, 2009] Amir Gershman, Amnon Meisels, and Roie Zivan. Asynchronous forward bounding for distributed COPs. *JAIR*, 34:61–88, 2009.

[Greenstadt *et al.*, 2007] R. Greenstadt, B. Grosz, and M. D. Smith. SSDPOP: improving the privacy of DCOP with secret sharing. In *AAMAS*, pages 171:1–171:3, 2007.

[Greenstadt, 2009] Rachel Greenstadt. An overview of privacy improvements to k-optimal DCOP algorithms. In *AAMAS*, pages 1279–1280, 2009.

[Grinshpoun and Tassa, 2014] T. Grinshpoun and T. Tassa. A privacy-preserving algorithm for distributed constraint optimization. In *AAMAS*, pages 909–916, 2014.

[Grinshpoun, 2012] T. Grinshpoun. When you say (DCOP) privacy, what do you mean? In *ICAART*, pages 380–386, 2012.

[Hirayama and Yokoo, 1997] K. Hirayama and M. Yokoo. Distributed partial constraint satisfaction problem. In *CP*, pages 222–236, 1997.

[Hirayama and Yokoo, 2005] K. Hirayama and M. Yokoo. The distributed breakout algorithms. *Artificial Intelligence*, 161:1-2:89–116, 2005.

[Katagishi and Pearce, 2007] H. Katagishi and J. P. Pearce. Kopt: Distributed DCOP algorithm for arbitrary k-optima with monotonically increasing utility. In *DCR*, 2007.

[Kiekintveld *et al.*, 2010] Christopher Kiekintveld, Zhengyu Yin, Atul Kumar, and Milind Tambe. Asynchronous algorithms for approximate distributed constraint optimization with quality bounds. In *AAMAS*, pages 133–140, 2010.

[Léauté and Faltings, 2013] Thomas Léauté and Boi Faltings. Protecting privacy through distributed computation in multi-agent decision making. *JAIR*, 47:649–695, 2013.

[Maheswaran *et al.*, 2004] R. T. Maheswaran, J. P. Pearce, and M. Tambe. Distributed algorithms for DCOP: A graphical-game-based approach. In *PDCS*, pages 432–439, 2004.

[Maheswaran *et al.*, 2006] Rajiv T. Maheswaran, Jonathan P. Pearce, and Milind Tambe. A family of graphical-game-based algorithms for distributed constraint optimization problems. In *Coordination of Large-Scale Multiagent Systems*, pages 127–146. Springer-Verlag, 2006.

[Modi *et al.*, 2005] P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. ADOPT: asynchronous distributed constraints optimizationwith quality guarantees. *Artificial Intelligence*, 161:149–180, 2005.

[Paillier, 1999] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt*, pages 223–238, 1999.

[Pearce and Tambe, 2007] J. P. Pearce and M. Tambe. Quality guarantees on k-optimal solutions for distributed constraint optimization problems. In *IJCAI*, pages 1446–1451, Hyderabad, India, January 2007.

[Petcu and Faltings, 2005] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *IJCAI*, pages 266–271, 2005.

[Sultanik *et al.*, 2008] E. Sultanik, R. N. Lass, and W. C. Regli. DCOPolis: a framework for simulating and deploying distributed constraint reasoning algorithms. In *AAMAS (demos)*, pages 1667–1668, 2008.

[Tassa and Gudes, 2012] T. Tassa and E. Gudes. Secure distributed computation of anonymized views of shared databases. *Transactions on Database Systems*, 37, Article 11, 2012.

[Tassa *et al.*, 2015] Tamir Tassa, Roie Zivan, and Tal Grinshpoun. Max-sum goes private. In *IJCAI*, pages 425–431, 2015.

[Teacy *et al.*, 2008] W. T. L. Teacy, A. Farinelli, N. J. Grabham, P. Padhy, A. Rogers, and N. R. Jennings. Max-Sum decentralised coordination for sensor systems. In *AAMAS*, pages 1697–1698, 2008.

[Vinyals *et al.*, 2011] M. Vinyals, E. A. Shieh, J. Cerquides, J. A. Rodríguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring. Quality guarantees for region optimal DCOP algorithms. In *AAMAS*, pages 133–140, 2011.

[Watts and Strogatz, 1998] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440–442, 1998.

[Zhang *et al.*, 2005] W. Zhang, Z. Xing, G. Wang, and L. Wittenburg. Distributed stochastic search and distributed breakout: properties, comparishon and applications to constraints optimization problems in sensor networks. *Artificial Intelligence*, 161:55–88, 2005.

[Zivan *et al.*, 2014] R. Zivan, S. Okamoto, and H. Peled. Explorative anytime local search for distributed constraint optimization. *Artificial Intelligence*, 212:1–26, 2014.