Coordinating Logistics Operations with Privacy Guarantees

Thomas Léauté and **Boi Faltings**
Artificial Intelligence Laboratory
École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
first_name.last_name@epfl.ch

Abstract

Several logistics service providers serve a certain number of customers, geographically spread over an area of operations. They would like to coordinate their operations so as to minimize overall cost. At the same time, they would like to keep information about their costs, constraints and preferences private, thus precluding conventional negotiation. We show how AI techniques, in particular Distributed Constraint Optimization (DCOP), can be integrated with cryptographic techniques to allow such coordination without revealing agents' private information. The problem of assigning customers to companies is formulated as a DCOP, for which we propose two novel, privacy-preserving algorithms. We compare their performances and privacy properties on a set of Vehicle Routing Problem benchmarks.

1 Introduction

Consider delivery companies that must serve customers geographically spread over an area of operations. Assume the companies are working under the same franchise, so that their common goal is to serve all customers at minimal total cost. While they desire to collaborate, the companies still want to protect their strategical information, such as their internal costs and constraints. This precludes the use of a fully centralized approach in which they would formalize their respective subproblems under a common framework, and report it to a third party that would perform the optimization. Furthermore, such a complete formalization of their subproblems might not even be feasible nor desirable, because each company is used to planning the routes of its own vehicle fleet using its own, specific planning tool. This problems calls for an integration of the companies' various vehicle routing tools into a multi-agent platform that coordinates their decisions.

In this paper, we propose to perform this integration using a master-slave problem decomposition approach, in which the master problem is the distributed coordination problem of assigning customers to companies, and the slave problems are the local Vehicle Routing Problems (VRPs) of the companies, as illustrated in Figure 1. Given a subset of customer demands that must be served, each company's VRP solver computes optimal routes for the company's vehicles, and outputs a corresponding optimal cost. These costs are fed to the master problem that consists in exploring the space of allocations of customer demands to companies, in search for the optimal allocation; this is where we propose to apply AI techniques, integrating them as one part of the larger software system.

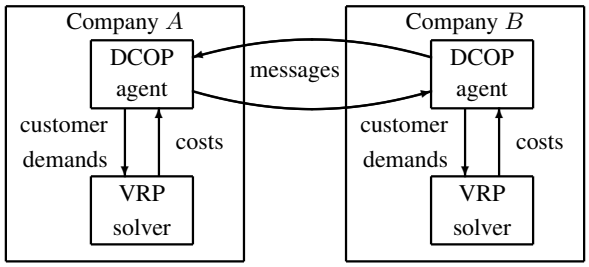

Figure 1: Integration model for the distributed VRP.

This paper focuses on the master problem, formalized as a Distributed Constraint Optimization Problem (DCOP). The algorithm should provide guarantees to the companies about the protection of their private information that could be leaked by the messages exchanged. To obtain such privacy guarantees, we investigate four DCOP algorithms, two of which are novel, which integrate various techniques borrowed from cryptography. We theoretically compare their privacy properties, and empirically compare their performances on VRP benchmarks, both in terms of distributed runtime, and amount of information exchanged between the companies.

One issue that we do *not* address, however, is that of manipulation of the DCOP algorithm by the companies so as to steer the solution towards one that better meets their respective selfish interests. Techniques based on payments can be used to address incentive-compatibility, by aligning each company's objective to the minimization of the overall cost; such techniques are outside the scope of this paper. We assume that each company truthfully reports to its DCOP agent the true optimal costs computed by its local VRP solver.

Section 2 first formalizes a distributed variant of the VRP, reviews some of the previous work on VRPs, and presents our general, DCOP-based approach to solve the master assignment problem while preserving the privacy of the partic-

ipants. Sections 3 and 4 then introduce two new, privacy-preserving DCOP algorithms for the master problem, and an empirical evaluation of their performances is reported in Section 5, in which we also compare against a novel variant of another algorithm based on Secure Multiparty Computation.

2 Preliminaries

Section 2.1 first introduces a new class of Vehicle Routing Problems, the DisSDMDVRP. Section 2.2 then briefly mentions the existing VRP literature, and Section 2.3 presents the DCOP framework we use to solve the problem of assigning customers to companies in the DisSDMDVRP.

2.1 Definition of the DisSDMDVRP

The *Distributed, Multiple-Depot, Vehicle Routing Problem (DisMDVRP)* was first defined by Léauté *et al.* [2010]. We propose a novel variant, in which we now allow any customer's demand to be split among the delivery companies.

Definition 1 (DisSDMDVRP) *A* Distributed, Split-Delivery, Multiple-Depot, Vehicle Routing Problem *consists of a set $D = \{d_1, \dots, d_{n_D}\}$ of depots in the Euclidian plane, each controlled by a different delivery company. Each company owns n_V vehicles, with maximum load $Q_{\max}$ and a maximum route length $L_{\max}$. The companies must serve a set $C = \{c^1, \dots, c^{n_C}\}$ of customers at known locations, so that each customer c_i has a demand $q_i \in \mathbb{N}$ that must be served fully, but can be split among companies. Each vehicle must come back to its initial depot at the end of its route. Each company also has a* visibility radius *$R \leq L_{\max}/2$ that defines the boundaries of its knowledge of the overall problem. The company owning depot d_i is only aware of the customers that are within distance R of d_i, and only knows another company if their areas of visibility overlap.*

The goal is for the companies to agree on who should serve which customers, using which vehicle routes, so as to serve all visible customers at minimal total route length.

The DisSDMDVRP can be seen as consisting of two subproblems (Figure 1). The slave problem is each company's VRP, given assignments of customer demands to depots; Section 2.2 briefly recalls methods that have been proposed to solve such problems. The master problem consists in optimally assigning customer demands to depots (Section 2.3).

2.2 Related Work on the Vehicle Routing Problem

There is a very large body of literature on the Vehicle Routing Problem and its numerous variants, traditionally named by appending pre- and suffixes to the *VRP* acronym; a rule from which we did not derogate in Definition 1. As the focus of this paper is not on solving these slave VRPs, we only briefly go over some of the previous work on this subject. For more comprehensive surveys of the VRP theory and practice, the reader can refer to [Hjorring, 1995; Toth and Vigo, 2001].

Examples of *complete* algorithms that have been proposed to solve VRPs to optimality include Branch-and-Bound search [Christofides and Eilon, 1969], Dynamic Programming [Christofides *et al.*, 1981], or Mixed-Integer Linear Programming techniques, such as Set Partitioning [Agarwal *et al.*, 1989] and Vehicle Flow [Laporte and Nobert, 1987].

However, due to the NP-hard nature of the problem, most algorithms proposed for the VRP are incomplete heuristics or metaheuristics. The most famous, specialized heuristics may be the *savings* algorithm by Clarke and Wright [1964], and the *sweep* algorithm by Gillett and Miller [1974]. More recently, metaheuristics [Gendreau *et al.*, 2007] were successfully applied to the VRP, such as Ant Colony Optimization, Genetic Algorithms, Simulated Annealing, or Tabu Search.

It is important to stress that the integrated approach we propose in this paper is not specific to any of these techniques. In fact, the delivery companies may use their own preferred VRP solvers, as long as the DCOP agent can query them for the optimal (or sub-optimal) costs of serving subsets of the customer demands (Figure 1).

2.3 Distributed Constraint Optimization

This section first describes the framework of Distributed Constraint Optimization (DCOP), and how we use it to model the problem of assigning customers to depots. We then present privacy properties expected from a DCOP algorithm, and an existing algorithm that exhibits some of these properties.

Definition 2 (DCOP) *A discrete* Distributed Constraint Optimization Problem *is a tuple $< \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C} >$, where:*

- *$\mathcal{A} = \{a_1, \dots, a_{|\mathcal{A}|}\}$ is a set of* agents*;*
- *$\mathcal{X} = \{X_1, \dots, X_n\}$ are* variables *owned by the agents;*
- *$\mathcal{D} = \{D_1, \dots, D_n\}$ is a set of finite* domains*, such that variable X_i takes values in $D_i = \{x_1, \dots x_k\}$;*
- *$\mathcal{C} = \{c_1, \dots, c_m\}$ is a set of soft* constraints*, where each c_i defines a* cost $\in \mathbb{R} \cup \{\infty\}$ *for each combination of assignments to a subset of variables. A constraint is initially known only to the agents involved.*

A solution *to the DCOP is an assignment to all variables that minimizes the overall sum of costs $\sum_i c_i$.*

Léauté *et al.* [2010] applied this formalism to the assignment problem for the DisMDVRP; we introduce the following novel adaptation that allows split deliveries, in which the agents are the depots. Depot d_i owns one variable $X_{i,j} \in [0, q_j]$ for each visible customer c^j of demand q_j, modeling how much of c^i's demand it serves. Variables need not be created for customers that can only be served by a single depot, since these variables would necessarily take their respective maximum values; this is an improvement over [Léauté *et al.*, 2010]. The DCOP constraints are of two types:

1. For each customer c^j, if the set D^j of depots within visible distance R of c^j is non-empty, then one constraint $\sum_{d_i \in D^j} X_{i,j} = q_j$ enforces that c^j must be fully served.
2. For each depot d_i, if the set $C_i = \{c^{j_1}, \dots, c^{j_n}\}$ of visible customers is non-empty, then one constraint $vrp_i(X_{i,j_1}, \dots, X_{i,j_n}) \in \mathbb{R} \cup \{\infty\}$ expresses the cost of the optimal solution to d_i's own VRP, as a function of how much of each customer's demand it serves.

The resulting constraint graph is illustrated in Figure 2, for a DisSDMDVRP instance taken from [Léauté *et al.*, 2010].

Faltings *et al.* [2008] defined four types of privacy guarantees one might expect from a DCOP algorithm; we recall them below, illustrating them on Figure 2.

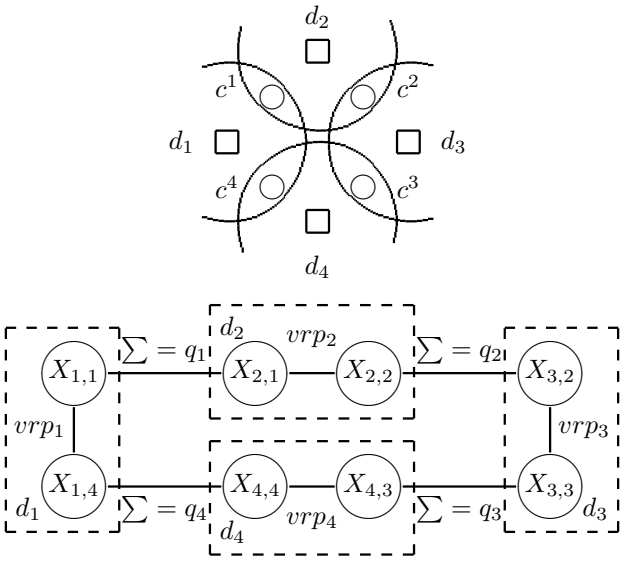

Figure 2: A simple DisSDMDVRP instance.

Agent privacy is respected if the algorithm does not reveal to any agent the identity nor even the presence of non-neighboring agents: company d_1 does not discover the existence of company d_3. This type of privacy guarantee might not be highly relevant if we assume that all delivery companies are working under the same franchise.

Topology privacy precludes agents from discovering topological constructs of the constraint graph (variables, constraint scopes, cycles...) they are not involved in. Depot d_1 should not discover anything about c^2 and c^3.

Constraint privacy covers the cost values of any constraint, which should not be revealed to any agent not involved in the constraint. Depot d_1 should not discover anything about the internal costs of the other depots, which includes for instance the details of their fleets of vehicles.[1]

Decision privacy imposes that no agent discovers the values chosen for variables it does not own. If a customer's demand is split among at least three companies, each only discovers how much of the demand it must serve, not how the remaining has been split among the others.

These four types of privacy guarantees necessarily exclude *semi-private information*, which is information that may be revealed by the solution chosen to the DCOP, regardless of the algorithm used to compute this solution. For instance, since customer c^1's demand q_1 can only be served by depots d_1 and d_2, then d_1 can inevitably infer how much of c^1's demand has been assigned to d_2, since the split must sum up to q_1.

Faltings *et al.* [2008] introduced the P-DPOP algorithm, which partially addresses these privacy guarantees as follows. Variables are first ordered along a *pseudo-tree* obtained by a distributed, depth-first traversal of the constraint graph. The choice of the root that initiates the traversal is performed using an *anonymous leader election* algorithm, guaranteeing the identity and position of the root in the constraint graph are only revealed to its owner agent. This pseudo-tree generation algorithm protects agent and topology privacy (Table 1).

[1]Definition 1 assumes that all companies have identical fleets, but this can be relaxed without making our approach invalid.

	agent	topology	constraint	decision
P-DPOP	✓	✓	∼	
P$^{3/2}$-DPOP	✓	✓	∼	✓
P^2-DPOP	✓	✓	✓	✓
MPC			∼	∼

Table 1: Privacy guarantees of various DCOP algorithms.

The leaves then initiate a bottom-up, cost propagation, during which costs are aggregated and variables are eliminated one after the other, along each branch of the pseudo-tree in parallel. Agents only send messages to neighbors, and variable names and values are replaced with secret codenames, so that agent and topology privacy are guaranteed. To address constraint privacy, secret, large, random numbers are added to the cost values in the messages in such a way that the variable elimination procedure can still be performed on obfuscated costs. This only guarantees *partial* constraint privacy (Table 1), because this obfuscation is not cryptographically secure; it leaks small amounts of statistical information about the cost values, which can be made as small as desired by increasing the sizes of the random numbers.

Once the cost propagation has reached the root, an optimal value for this variable is obtained, and sent back down the pseudo-tree until all variables have been assigned optimal values. During this last phase, each agent discovers the values of its ancestor variables in the pseudo-tree, and therefore decision privacy is partially violated (Table 1).

3 The P^2-DPOP Algorithm for DCOP

To address the privacy leaks in the P-DPOP algorithm, Léauté and Faltings [2009] proposed P^2-DPOP (Section 3.1). Their algorithm only applies to *Distributed Constraint Satisfaction Problems (DisCSPs)*, which are restrictions of DCOPs in which costs can only take values in $\{0, \infty\}$. Section 3.2 introduces a novel generalization to DCOPs.

3.1 The P^2-DPOP Algorithm for DisCSP

Léauté and Faltings [2009] replaced P-DPOP's obfuscation method by ElGamal homomorphic encryption [Elgamal, 1985]. Encryption is performed using one public key, and decryption is collaborative, requiring all agents to use their respective private keys. By representing the cost value ∞ with the cleartext integer 1, and the cost value 0 with a cleartext integer $z \neq 1$, the multiplicative homomorphic property enables the minimization, with respect to some variable X_i, of an encrypted cost function $E(c(X_i))$, as follows:

$$\min_{X_i = x_1 \ldots x_n} E(c(X_i)) = E(c(x_1)) \times \ldots \times E(c(x_n)) \quad (1)$$

$$= E(c(x_1) \times \ldots \times c(x_n)) \,, \quad (2)$$

which decrypts to a cost of ∞ if and only if $\forall j\ c(x_j) = 1$ (i.e. $c(X_i)$ is constantly equal to ∞), and to a cost of 0 otherwise.

However, ElGamal encryption is not additively homomorphic, so when it comes to summing costs, it is only possible to add an encrypted cost $E(c)$ with a cleartext cost $\in \{0, \infty\}$, which is done as follows:

$$E(c) + 0 = E(c) \qquad E(c) + \infty = E(1)\ .$$

This *partial* homomorphism is the reason why it is necessary to adopt a distributed algorithm, in which each company locally adds its cleartext costs to encrypted costs received from other companies. A centralized approach in which all computation would be performed centrally on encrypted costs would require *fully homomorphic* encryption. Such encryption schemes were recently proposed, such as in [Gentry, 2009], but they are impractical, and currently do not scale to DisSDMDVRP instances of any reasonable sizes.

Another consequence of this limitation is that the pseudo-tree must be constructed such that any variable has at most one child, i.e. the pseudo-tree contains a single branch. Such a single-branch pseudo-tree does not necessarily exist without introducing parent-child relationships between non-neighboring variables, which is a threat to agent and topology privacy. To address this issue, Léauté and Faltings [2009] describe a message routing procedure that is used during the bottom-up propagation of costs along the pseudo-tree, such that agent and topology privacy remain guaranteed.

Finally, the loss of decision privacy in P-DPOP is addressed by eliminating the final, top-down propagation of variable assignments, and instead repeating the bottom-up cost propagation, with each variable at the root of the pseudo-tree in turn. The resulting P^2-DPOP algorithm has better privacy properties than P-DPOP (Table 1), but this comes at a high price in terms of performance, as shown in Section 5.

3.2 Adding Support for Optimization

As explained in the previous section, Léauté and Faltings' P^2-DPOP algorithm [2009] is only applicable to DisCSPs, in which the costs are restricted to taking values in $\{0, \infty\}$. In this section, we provide a novel generalization to DCOPs, with integer cost values in $[0, c_{\max}] \cup \{\infty\}$, where $c_{\max} \in \mathbb{N}$ is a known upper bound on the cost of the optimal solution. A cost c is represented with the following cleartext vector:

$$c \rightarrow [\overbrace{1, \ldots, 1}^{c}, \overbrace{z, \ldots, z}^{c_{\max}+1-c}]\ .$$

Cost values greater than $c_{\max}$ (including ∞) are represented with vectors full of 1's, and cannot be discerned from ∞.

Computing the $\min$ of encrypted costs is performed by applying the same operation as in Eq. (1), element-wise on the vectors. The sum of an encrypted cost with a cleartext cost c is computed by shifting the vector by c as follows:

$$[E_0, \ldots, E_{c_{\max}}] + c = [\overbrace{E(1), \ldots, E(1)}^{c}, E_0, \ldots, E_{c_{\max}-c}]\ .$$

This generalization of P^2-DPOP only increases the runtime and message sizes of the first cost propagation phase by a factor of $(c_{\max} + 1)$. At the end of this first propagation, the agents discover the cost c_{opt} of the optimal solution, and can use smaller vectors of size $(c_{opt} + 1)$ for the subsequent cost propagations. We propose, as an additional improvement to the P^2-DPOP algorithm, to interrupt the algorithm after the first propagation phase, if the optimal cost has been found equal to ∞. This possibility of early termination on infeasible problems was not mentioned by Léauté and Faltings [2009].

4 The P$^{3/2}$-DPOP Algorithm

As will be shown empirically in Section 5, the cost in terms of performance that P^2-DPOP has to pay to patch the two privacy leaks in P-DPOP is significant. The two main sources of this added complexity are the following:

1. The use of single-branch pseudo-trees imposed by ElGamal encryption produces variable orderings of larger *induced widths*, and Petcu and Faltings [2005] have shown that the largest cost message is exponential in this parameter. The cost messages in P^2-DPOP therefore contain on average more cost values than in P-DPOP, and, on top of this, encrypting each cost value is more expensive, because ElGamal encryption costs more than simple obfuscation by addition of large random numbers.

2. A factor of less influence on performance is the fact that P^2-DPOP does not perform only one bottom-up cost propagation, but one per variable (except on infeasible problems), each variable being at the root of the pseudo-tree in turn. This inherently multiplies the complexity by the number of variables in the problem. Furthermore, incrementally changing the root of the pseudo-tree is also itself expensive, because it must be done in a way that preserves agent and topology privacy. The method proposed by Léauté and Faltings [2009] also internally uses expensive ElGamal operations to preserve privacy.

If one analyses these two changes in P^2-DPOP in terms of "return on investment," one can clearly see that the first, very expensive change only brings a minor improvement to constraint privacy, which was already partially guaranteed in P-DPOP (Section 2.3 and Table 1). On the other hand, the second, less expensive change guarantees decision privacy, which is violated in P-DPOP.

Based on this observation, we propose a novel algorithm that is a hybrid of P-DPOP and P^2-DPOP, called P$^{3/2}$-DPOP, which guarantees decision privacy using the latter technique, but does not use the former to obtain full constraint privacy (Table 1). We claim that, in many real-life situations, the partial constraint privacy guarantees given by P-DPOP and P$^{3/2}$-DPOP can be considered satisfactory, and the use of expensive homomorphic encryption is an overkill. In fact, while obfuscation does leak minor amounts of statistical information about constraint costs, one could argue that it is actually stronger than ElGamal encryption, as it does not rely on the worst-case computational complexity of breaking private keys. Section 5 shows empirically that P$^{3/2}$-DPOP performs significantly better than P^2-DPOP, and sometimes only marginally worse than P-DPOP.

prob.	n_D	R	n_C	$q_{\max}$	P-DPOP		$P^{3/2}$-DPOP		P^2-DPOP		MPC-DisWCSP4	
					time	info	time	info	time	info	time	info
p01	4	13	2 / 27	25	395.0 ms	10 kB	582.0 ms	25 kB	26.5 s	10 MB	-	-
		14	4 / 30	25	2.1 s	27 kB	22.2 s	779 kB	-	-	-	-
p03	2	10	1 / 14	20	462.0 ms	5 kB	*756.0 ms*	*12 kB*	11.8 s	2 MB	*25.2 s*	*8 MB*
		12	2 / 16	20	2.1 s	21 kB	*4.4 s*	*98 kB*	5.0 min	39 MB	*6.2 min*	*130 MB*
p11	2	22	1 / 19	47	1.2 s	9 kB	*1.9 s*	*20 kB*	49.8 s	10 MB	*2.1 min*	*69 MB*
		24	2 / 23	47	11.3 s	80 kB	*37.2 s*	*334 kB*	51.6 min	372 MB	-	-
p12	2	65	2 / 72	1	324.0 ms	3 kB	*690.0 ms*	*23 kB*	1.5 min	10 MB	*1.5 min*	*414 MB*
		70	4 / 72	2	879.0 ms	7 kB	*5.0 s*	*110 kB*	31.5 min	191 MB	*11.1 min*	*2 GB*
		79	8 / 80	2	6.1 s	58 kB	*5.2 min*	*2 MB*	-	-	-	-
		80	10 / 80	4	2.1 min	1 MB	-	-	-	-	-	-
p15	4	60	8 / 144	1	943.0 ms	19 kB	13.0 s	1 MB	-	-	-	-
		70	16 / 144	2	9.3 min	17 MB	-	-	-	-	-	-
p18	6	60	14 / 215	1	2.6 s	86 kB	2.2 min	7 MB	-	-	-	-
p21	9	60	24 / 324	1	34.4 s	2 MB	-	-	-	-	-	-

Table 2: Experimental results on Cordeau MDVRP benchmarks. n_D is the number of depots that must coordinate their decisions, and n_C is reported as *number of customers that can be served by at least two depots / total number of visible customers*. To solve each problem instance, P^2-DPOP and MPC-DisWCSP4 need an upper bound $c_{\max}$ respectively on the optimal cost and on the worst feasible cost; this bound was set to the sum of the worst costs of all VRPs.

5 Experimental Results

Experiments were performed on the open-source FRODO platform for DCOP [Léauté *et al.*, 2009]. DisSDMDVRP instances were generated based on the Cordeau MDVRP benchmark instances from [VRP Web, 2007], by varying the visibility radius R of the depots. We reused the same problem instances p01, p03 and p11 as in [Léauté *et al.*, 2010] (except that we are now allowing split deliveries), and we also considered four other instances p12, p15, p18 and p21, which involve more depots and more customers, but smaller maximum demands $q_{\max}$. Two performance metrics were used: the simulated time [Sultanik *et al.*, 2007], and the total amount of information exchanged by the agents. We do not report solution quality, being the same for all algorithms. The experiments were run on a 2.53-GHz computer, with 2 GB of Java heap space, and a timeout of 1 hour (wall clock time). Each entry in Table 2 is the median over 43 runs.

To solve each depot's local VRP, the FRODO platform was coupled with the OR-Objects Library [OpsResearch, 2010], which was recently made open-source. The VRP algorithm was set to the best out of Clarke and Wright's *savings* algorithm [1964] and Gillett and Miller's *sweep* algorithm [1974] as construction algorithm, with the 2-Opt TSP improvement algorithm [Croes, 1958].

As a baseline for comparison with P-DPOP, $P^{3/2}$-DPOP and P^2-DPOP, we also implemented the MPC-DisWCSP4 algorithm, which is based on the MPC-DisCSP4 algorithm [Silaghi, 2005] for DisCSPs, with the *weak extension to weighted DisCSPs (DisWCSPs)* described for the earlier MPC-DisCSP2 algorithm in [Silaghi and Mitra, 2004]. Its privacy properties are summarized in Table 1. First, it violates agent privacy, because it requires all agents to be able to communicate pairwise securely. Second, it also violates topology privacy, because it requires each agent to know all variables in the problem, and their respective domains. In fact, it was only designed to protect constraint and decision privacy, but even on this front, its privacy properties are limited by the fact that it relies on Shamir's secret sharing scheme [1979], which is a $\lfloor \frac{n_D}{2} \rfloor$-threshold scheme. This means that if only $\lfloor \frac{n_D}{2} \rfloor$ depots collude, they can break all other depots' constraint and decision privacy. This is particularly problematic for the problem instances involving $n_D < 4$ depots, because the collusion threshold is then $\lfloor \frac{n_D}{2} \rfloor = 1$, which means that no privacy is guaranteed whatsoever.

These problem instances are indicated in italics in Table 2, and happen to be the only ones that MPC-DisWCSP4 could solve before the timeout. On these instances, P^2-DPOP had comparable performance in terms of runtime, but exchanged up to 40 times less information. Furthermore, where MPC-DisWCSP4's threshold scheme failed to guarantee any level of privacy, P^2-DPOP had a perfect score (Table 1). P^2-DPOP was also able to solve more instances.

Table 2 also shows that replacing ElGamal encryption with obfuscation by addition of random numbers, like in P-DPOP and $P^{3/2}$-DPOP, cut the runtime by 1 to 2 orders of magnitude, and the information exchange by 3 orders of magnitude, making it possible to solve many more instances. In terms of constraint privacy guarantees, these performance improvements only come at a minor loss of privacy.

Finally, comparing $P^{3/2}$-DPOP against P-DPOP shows that decision privacy (Table 1) can often be achieved at low performance losses. It is only when the number of customers on which depots need to coordinate (first number in column n_C) gets larger, that $P^{3/2}$-DPOP's performance starts to degrade significantly. This is consistent with the theoretical performance analysis in Section 4. Notice however that, on problem instances with only 2 depots (in italics), decision privacy actually cannot be guaranteed, since all decisions are semi-private information that cannot be protected (Section 2.3).

6 Conclusion

In this paper, we have presented a method to address Vehicle Routing Problems (VRPs) involving multiple delivery companies, which integrates the companies' possibly heterogenous route planners into a multi-agent platform that optimally coordinates their decisions. Special emphasis has been put on solving the coordination problem of assigning customer demands to companies, which we have proposed to model as a Distributed Constraint Optimization Problem (DCOP).

Our goal being to protect the privacy of the companies' information, we have proposed two novel DCOP algorithms that provide various levels of privacy guarantees, by integrating techniques borrowed from the field of cryptography. Our experimental evaluation on VRP benchmarks have shown that reasonable levels of privacy can be achieved for medium-sized problems, at acceptable performance costs in terms of computational runtime and information exchange.

The issue of incentive-compatibility has been left outside the scope of this paper. By selectively decrypting the relevant entries in the cost messages, it is possible to determine the cost each agent claimed and thus implement a first-price payment scheme where each agent gets paid this value. While such schemes are the norm in auctions today, they are not truthful. It is also possible to implement a truthful second-price scheme by analyzing the marginal cost for each variable, but this requires significant additional computation that could affect the privacy properties. Furthermore, second-price schemes are vulnerable to well-known problems such as false-name bidding and revenue failures.

References

[Agarwal *et al.*, 1989] Y. Agarwal, K. Mathur, and H. M. Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19:731–749, 1989.

[Christofides and Eilon, 1969] N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *Operational Research Quarterly*, 20:309–318, 1969.

[Christofides *et al.*, 1981] N. Christofides, A. Mingozzi, and Paolo Toth. Space state relaxation procedures for the computation of bounds to routing problems. *Networks*, 11:145–164, 1981.

[Clarke and Wright, 1964] G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4):568–581, July–August 1964.

[Croes, 1958] G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, November–December 1958.

[Elgamal, 1985] Taher Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. on Information Theory*, 31(4):469–472, July 1985.

[Faltings *et al.*, 2008] Boi Faltings, Thomas Léauté, and Adrian Petcu. Privacy Guarantees through Distributed Constraint Satisfaction. In *IAT'08*, pages 350–358, 2008.

[Gendreau *et al.*, 2007] Michel Gendreau, Jean-Yves Potvin, Olli Bräysy, Geir Hasle, and Arne Løkketangen. Metaheuristics for the vehicle routing problem and its extensions : A categorized bibliography. Technical Report 2007-27, CIRRELT, August 2007.

[Gentry, 2009] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proc. 41st Annual ACM Symp. on Theory of Computing (STOC'09)*, pages 169–178, 2009.

[Gillett and Miller, 1974] Billy E. Gillett and Leland R. Miller. A heuristic algorithm for the vehicle-dispatch problem. *Ops Research*, 22(2):340–349, March–April 1974.

[Hjorring, 1995] C. Hjorring. *The Vehicle Routing Problem and Local Search Metaheuristics*. PhD thesis, Dept of Engineering Science, The University of Auckland, 1995.

[Laporte and Nobert, 1987] Gilbert Laporte and Y. Nobert. Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics*, 31:147–184, 1987.

[Léauté and Faltings, 2009] Thomas Léauté and Boi Faltings. Privacy-Preserving Multi-agent Constraint Satisfaction. In *Proc. IEEE International Conference on PrivAcy, Security, riSk And Trust (PASSAT'09)*, pages 17–25, 2009.

[Léauté *et al.*, 2009] Thomas Léauté, Brammert Ottens, and Radoslaw Szymanek. FRODO 2.0: An Open-Source Framework for Distributed Constraint Optimization. In *Proceedings of the DCR'09 Workshop*, pages 160–164, 2009. `http://liawww.epfl.ch/frodo`.

[Léauté *et al.*, 2010] Thomas Léauté, Brammert Ottens, and Boi Faltings. Ensuring Privacy through Distributed Computation in Multiple-Depot Vehicle Routing Problems. In *Proceedings of the ECAI'10 Workshop on Artificial Intelligence and Logistics (AILog'10)*, pages 25–30, 2010.

[OpsResearch, 2010] Operations Research – Java Objects. `http://or-objects.org/`, 2010.

[Petcu and Faltings, 2005] Adrian Petcu and Boi Faltings. DPOP: A Scalable Method for Multiagent Constraint Optimization. In *Proc. IJCAI'05*, pages 266–271, 2005.

[Shamir, 1979] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[Silaghi and Mitra, 2004] Marius-Călin Silaghi and Debasis Mitra. Distributed constraint satisfaction and optimization with privacy enforcement. In *Proc. Intl Conf. on Intelligent Agent Technology (IAT'04)*, pages 531–535, 2004.

[Silaghi, 2005] Marius-Călin Silaghi. Hiding absence of solution for a distributed constraint satisfaction problem (poster). In *Proc. of the 18th Intl Florida A.I. Research Society Conference (FLAIRS'05)*, pages 854–855, 2005.

[Sultanik *et al.*, 2007] Evan A. Sultanik, Robert N. Lass, and William C. Regli. DCOPolis: A framework for simulating and deploying distributed constraint optimization algorithms. In *Proceedings of the DCR'07 Workshop*, 2007.

[Toth and Vigo, 2001] Paolo Toth and Daniele Vigo, editors. *The Vehicle Routing Problem*. SIAM, 2001.

[VRP Web, 2007] The VRP Web. `http://neo.lcc.uma.es/radi-aeb/WebVRP/`, March 2007.