Journal of Artificial Intelligence Research 64 (2019) 987-1023 Submitted 12/18; published 04/19

COOPT: Using Collective Behavior of Coupled Oscillators for Solving DCOP

Allan R. Leite ALLAN@PPGIA.PUCPR.BR
Department of Applied Informatics
Pontifical Catholic University of Parana
Imaculada Conceição St., 1155, Curitiba-PR, Brazil

Fabrício Enembreck FABRICIO@PPGIA.PUCPR.BR
Department of Applied Informatics
Pontifical Catholic University of Parana
Imaculada Conceição St., 1155, Curitiba-PR, Brazil

Abstract

The distributed constraint optimization problem (DCOP) has emerged as one of the most promising coordination techniques in multiagent systems. However, because DCOP is known to be NP-hard, the existing DCOP techniques are often unsuitable for large-scale applications, which require distributed and scalable algorithms to deal with severely limited computing and communication. In this paper, we present a novel approach to provide approximate solutions for large-scale, complex DCOPs. This approach introduces concepts of synchronization of coupled oscillators for speeding up the convergence process towards high-quality solutions. We propose a new anytime local search DCOP algorithm, called Coupled Oscillator OPTimization (COOPT), which amounts to iteratively solving a DCOP by agents exchanging local information that brings them to a consensus. We empirically evaluate COOPT on constraint networks involving hundreds of variables with different topologies, domains, and densities. Our experimental results demonstrate that COOPT outperforms other incomplete state-of-the-art DCOP algorithms, especially in terms of the agents' communication cost and solution quality.

1. Introduction

The distributed constraint optimization problem (DCOP) provides a powerful formalism for modeling many decentralized coordination tasks in multiagent systems, and is widely used in realistic applications, such as sensor networks (Farinelli, Rogers, & Jennings, 2014; Rust, Picard, & Ramparany, 2016), task scheduling (Maheswaran, Tambe, Bowring, Pearce, & Varakantham, 2004b; Leite, Giacomet, & Enembreck, 2009), resource allocation (Carpenter et al., 2007; Amigoni, Castelletti, & Giuliani, 2015), among others. However, because DCOP has been shown to be an NP-Hard problem (Modi, Shen, Tambe, & Yokoo, 2005), some applications are unsuitable for applying this formalism, owing to a lack of efficient DCOP algorithms especially on hard instances of DCOP involving large-scale, complex domains.

Synchronization is a natural physical phenomenon that characterizes the spontaneous emergence of collective behavior in many complex dynamical systems (Strogatz, 1997). Mathematical models of synchronization in coupled oscillator networks have been intensely studied in recent decades because of the robustness and applicability of such models in large-scale and real-world domains (Buck & Buck, 1976; Sherman, Rinzel, & Keizer, 1988; Strogatz, 2003; Dörfler, M., & F., 2013). Moreover, such models are commonly flexible enough to be adapted to different contexts and also are able to support a wide variety of synchronization patterns, including optimization techniques as well as complex interaction dynamics (Acebrón, Bonilla, Pérez-Vicente, Ritort, & Spigler, 2005).

The contributions of this study are twofold. First, we investigate synchronization models as a new foundation to propel the evolution of DCOP research. Second, we introduce a new anytime local search algorithm designed to handle large-scale DCOPs, called Coupled Oscillator OPTimization (COOPT). Our algorithm is inspired by the collective behavior of coupled phase oscillators described by the Kuramoto model (Kuramoto, 1975). The Kuramoto model displays the synchronization dynamics in coupled oscillator networks from sinusoidal couplings in which each oscillator gradually adjusts its frequency to a mean-field by updating its phase after receiving new pulses from its neighborhood.

The remainder of this paper is organized as follows. Section 2 gives a brief overview of DCOP background. In Section 3, related works are considered. Section 4 reviews concepts of dynamical systems and mathematical models of synchronization in coupled oscillators. In Section 5, we introduce COOPT, a new anytime local search DCOP algorithm based on the synchronization dynamics of coupled oscillators. We present an empirical evaluation of COOPT by comparing it with other incomplete state-of-the-art DCOP algorithms in Section 6. Finally, in Section 7 we draw conclusions about the COOPT algorithm and discuss future works.

2. DCOP Background

By definition, a DCOP is a tuple $< \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{R}, \alpha >$, where $\mathcal{R}$ is a set of constraints and $\mathcal{X}$ is a set of variables distributed among a set $\mathcal{A}$ of agents. Each variable $x_i \in \mathcal{X}$ is held by a single agent in $\mathcal{A}$ and has a finite and discrete domain $D_i \in \mathcal{D}$. Such mapping from variables to agents determines the control $\alpha(x_i)$ of each variable $x_i \in \mathcal{X}$ to an agent. Thus, each value $d \in D_i$ represents one of possible states of a given agent. The constraints are defined by a set $\mathcal{R}$ of cost or reward relations between a pair of variable assignments.

Typically, binary cost functions are considered in DCOP formulations. A binary cost function for a pair of variables x_i and x_j is defined by $f_{ij} : D_i \times D_j \rightarrow \mathbb{Z}^+ \cup \{0\}$. Two agents are neighbors if there exists a constraint between them. Therefore, DCOP aims to find a complete set of A^* assignments, where $A^* = \{d_1, ..., d_n \mid d_1 \in D_1, ..., d_n \in D_n\}$, such that the global objective function $F(A)$ is minimized according to:

$$F(A) = \sum_{x_i, x_j \in \mathcal{X}} f_{ij}(d_i, d_j) \tag{1}$$

$$A^* = \underset{A \in \mathcal{S}}{\operatorname{argmin}}\ F(A), \tag{2}$$

when $x_i \leftarrow d_i$ and $x_j \leftarrow d_j$ in A (Fioretto, Pontelli, & Yeoh, 2018). Equation 1 represents $f_{ij} \in \mathcal{R}$ as a factor of the global objective function. Equation 2 refers to $\mathcal{S}$ as the state space for a given problem.

A DCOP structure is usually represented as a graph, where the variables correspond to vertexes and the constraints between pairs of variables are represented by the edges.

Representing a DCOP as a graph allows algorithms to perform searches or inferences from a local context taking into account constraints with neighboring agents. Therefore, such local computations enable agents to explore independent graph regions simultaneously by using a distributed problem-solving method (Lesser, Ortiz, & Tambe, 2003).

In recent years, there has been increasing interest in developing new DCOP algorithms (Leite, Enembreck, & Barthès, 2014; Fioretto et al., 2018). Complete algorithms are able to find optimal solutions, however they require exponentially increasing computational or communication resources with respect to the number of agents and variables, domain size of the variables, or constraints. On the other hand, incomplete algorithms can provide suboptimal solutions faster than complete algorithms. Nevertheless, large-scale, complex applications remain a challenge even for incomplete algorithms, especially those involving hundreds of variables or thousands of constraints.

3. Related Work

The Distributed Breakout Algorithm (DBA) and Distributed Stochastic Algorithm (DSA) (Zhang, Wang, Xing, & Wittenburg, 2005) are the two most popular incomplete methods for solving DCOP. Both algorithms perform local searches, in which agents make iteratively decisions on whether to change the assignments of their variables. Each agent begins the search by selecting a value for its variable randomly. Next, agents propagate the selected values and gather the values chosen by neighboring agents. For DSA, agents select values that improves its local cost, considering a probability p. In contrast, DBA manipulates the weights of the DCOP constraints in an attempt to avoid local optima.

Another popular incomplete DCOP algorithm is the Maximum Gain Message (MGM), which is an extension of DBA (Maheswaran, Pearce, & Tambe, 2004a). Each agent begins the search by selecting a value for its variable randomly. Agents propagate the selected values and gather the values chosen by neighboring agents. Next, each agent selects a value that results in a better unilateral gain and notifies neighboring agents about such gain. Then the agent replaces its current value if the gain is greater than the maximum gain of all neighboring agents.

In fact, DBA, DSA, and MGM explore only local optimization; that is, no unilateral change can improve the solution quality. Pearce, Tambe, and Maheswaran (2008) proposed an extension of MGM based on the k-optimality concept, called the Maximum Gain Message (MGM-2). The k-optimality concept refers to the formation of groups of agents such that no group with k or fewer agents are able to improve the current solution. Thus, MGM-2 is able to guarantee sub-optimal solutions within a distance $k = 2$.

Other relevant extension of local search algorithm is the Distributed Asynchronous Local Optimization (DALO) (Kiekintveld, Yin, Kumar, & Tambe, 2010). DALO uses k-optimality and t-distance concepts for providing a mechanism to coordinate the decisions of local groups of agents, rather than each agent performing individual choices. The t-distance considers a predefined distance from the central agent in the group. In contrast, DBA, DSA, and MGM are considered essentially 1-optimal and 0-distance. An alternative approach which combines both k-optimality and t-distance concepts, called $\mathcal{C}$-optimality, was recently introduced (Vinyals et al., 2011).

Most incomplete DCOP algorithms are in fact search-based or inference-based methods. Ottens, Dimitrakakis, and Faltings (2017) proposed a notable approach for solving DCOP using sampling techniques. The Distributed Upper Confidence Tree (DUCT) introduced confidence bounds in order to explore the solution search. DUCT constructs a confidence bound B, such that the best value for any context is at least B, and sample the choice with the lowest bound. In addition, Nguyen, Yeoh, and Lau (2013) proposed a memory-bounded sampling-based algorithm, called Distributed Gibbs (D-Gibbs).

Zivan, Okamoto, and Peled (2014) designed a generic framework that ensures an anytime property on local search algorithms; that is, the algorithm always return the best global solution explored when it ends up. This framework, called *Anytime Local Search DCOP* (ALS_DCOP), provides a communication model for anytime local search from a spanning tree on the constraint graph. Authors also demonstrated the effectiveness of this general framework in a new algorithm, named DSA Slope Dependent Probability (DSA-SDP), which employs a new heuristic for calculating the probability of replacing the assignment.

Other recent contribution about local search algorithms is an extension of DBA, called Generalized DBA (GDBA) (Okamoto, Zivan, & Nahon, 2016). DBA originally increases the weights of constraints whenever breakouts are detected. GDBA suggests three design choices of replacing the variable assignments. First, modifying weights of constraints as in original DBA, but in different manners (multiplicative or additive). Second, using definitions of constraint violation (non-zero, non-minimum and maximum constraint violation). Third, the scope of changes specifies which weights increase when breakouts are performed for a violated constraint.

4. Synchronization in Coupled Oscillators

Models of dynamical systems aim to describe components that evolve over time following a certain function or rule. This evolution occurs in an iterative way from the current state of the components, successively producing new states. Synchronization is one of the most important research areas of nonlinear dynamical systems, which aims to understand the trends of synchronously operating systems (Pikovsky, Rosenblum, & Kurths, 2001).

4.1 Kuramoto Model

One of the most successful attempts to understand and reproduce the synchronization phenomenon is known as the Kuramoto model (Kuramoto, 1975). The Kuramoto model exhibits the emergence of collective behavior in a coupled phase oscillator network. An oscillator is a device that autonomously produces repeated signals. In particular, phase oscillators have constant amplitudes, in which each phase value represents the state of an oscillator at a given time. The space-state of an oscillator can be defined by a circular interval $[0, 2\pi]$, where $0 = 2\pi$. Each oscillator emits a pulse to its neighboring oscillators whenever its phase reaches a certain threshold (Acebrón et al., 2005).

Formally, the Kuramoto model consists of a population of N globally coupled phase oscillators. Each oscillator i has an individual phase θ_i and a natural frequency ω_i. The Kuramoto model captures the emergence of the collective behavior in coupled oscillator network by gradual adjustments of the rhythms (frequencies) of the oscillators toward a

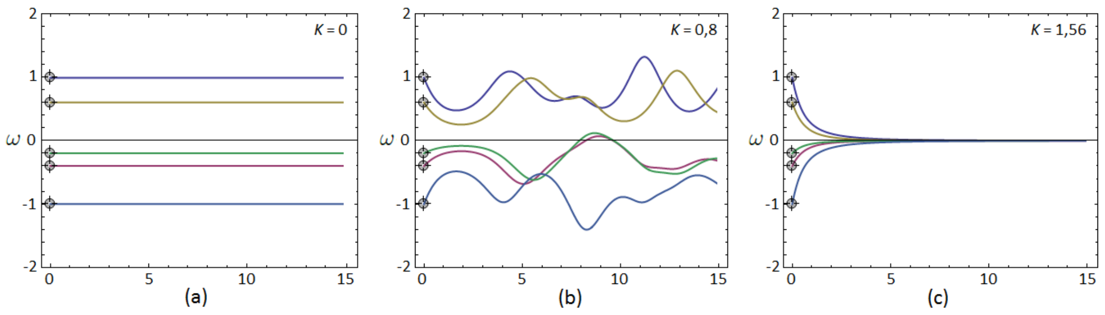


Figure 1: Synchronization dynamics of a globally coupled oscillator network. Generated from Mathematica Software (Wolfram, 2003).

mean-field represented by Ω (Kuramoto, 1975). These adjustments occur iteratively when oscillators receive new pulses from neighboring oscillators through sinusoidal couplings.

In this sense, such collective dynamics of the oscillator network make the phases become more similar over time, governed by Equation 3 (Kuramoto, 1984):

$$\frac{d\theta_i}{d_t} = \omega_i + \frac{K}{N}\sum_{j=1}^{N} \sin(\theta_j - \theta_i), \tag{3}$$

where K is the global coupling strength and N is the number of oscillators in the network. The frequencies ω_i are usually represented by some distribution function $g(\omega)$. When the frequency distribution is too dissimilar with respect to the coupling strength, the oscillators are unable to synchronize their phases and hence the system behaves incoherently over time. Therefore, a coupled oscillator network is able to reach a phase-synchronized state if there exists a coupling strength K enough regarding the heterogeneity of the natural frequencies of the oscillators.

Synchronization can also be represented in a more convenient way from the following complex-valued order parameter, according to Equation 4 (Kuramoto, 1984):

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\theta_j}, \tag{4}$$

where $0 \leq r(t) \leq 1$ measures the coherence level among all oscillators and $\psi(t)$ is the average phase of the system. The values $r(t) \approx 0$ and $r(t) \approx 1$ indicate absence of synchronization and the system being in a phase-synchronized state, respectively.

Figure 1 shows the synchronization dynamics in a coupled oscillator network, focusing on the coupling strength dominance over the natural frequencies. This example has five globally coupled oscillators with different natural frequencies. In Figure 1.a, the coupling K is null, and thus the oscillators preserve their natural frequencies ω_i over time. In Figure 1.b, there is a coupling $K = 0.8$, but this coupling strength is insufficient to reach a phase-synchronized equilibrium. In this case, the phases evolve incoherently. Finally, Figure 1.c shows a coupling strength $K = 1.56$, which is sufficient to reach a phase-synchronized equilibrium; that is, all the phases will be aligned after a certain time.

4.2 Synchronization in Complex Networks

A coupled oscillator network can be represented by a graph $G(\mathcal{V}, \mathcal{E})$ composed by vertexes $\mathcal{V} = \{1, ..., N\}$ and edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, where the vertexes represent oscillators and edges represent couplings between pairs of oscillators. The edges $\{i, j\} \in \mathcal{E}$ can be represented by an adjacency matrix a, where K is distributed among weighted edges $a_{ij} > 0$. In a typical globally coupled oscillator network, all edges share the same coupling strength, $\forall \{i, j\} \in \mathcal{E} \mid a_{ij} = \frac{K}{N}$. Nevertheless, complex networks usually do not have a coupling pattern; that is, $\exists \{i, j\} \in \mathcal{E} \mid a_{ij} = 0$. Moreover, complex networks often do not have homogeneous coupling strength.

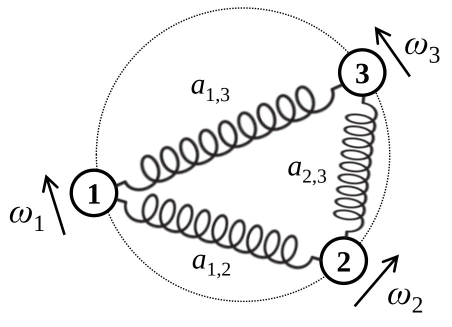


Figure 2: Example of a complex coupled oscillator network.

Figure 2 depicts an example of a complex oscillator network with heterogeneous coupling strength, in which the oscillators rotate around the same path without colliding. Each oscillator has a phase angle θ_i and a preferred natural rotation frequency ω_i representing the rotation speed. Pairs of coupled oscillators $\{i, j\}$ are connected by elastic springs with different strengths a_{ij}. Hence, the movements of a given oscillator i are influenced by the couplings a_{ij} with neighboring oscillators, because these connections have sufficient coupling strength to interfere in the frequency ω_i.

Originally the Kuramoto model displays the synchronization dynamics in a globally connected network; that is, a particular case of regular networks. However, several real-world application and complex systems can exhibit different connection patterns or network topologies. So a flexible synchronization model should be able to handle different connection patterns in order to be applicable to many classes of real-world applications.

Figure 3 illustrates four examples of different network topologies, whose vertexes can have different degrees following certain connection patterns. In regular networks, all vertexes have the same degree (Wasserman & Faust, 1994). On the contrary, random networks (Erdös & Rényi, 1959) suggest the idea of disordered networks, in which the connections between vertexes occur randomly. A random network may be represented as a graph in which connections between each pair of vertexes occur with probability p.

Another popular model widely adopted for describing complex networks is known as small-world effect (Watts & Strogatz, 1998). A small-world network can be represented by a graph with a starting lattice of dimension d and probability p of rewiring the connections, producing connected clusters. Thus, small-world and random networks have essentially

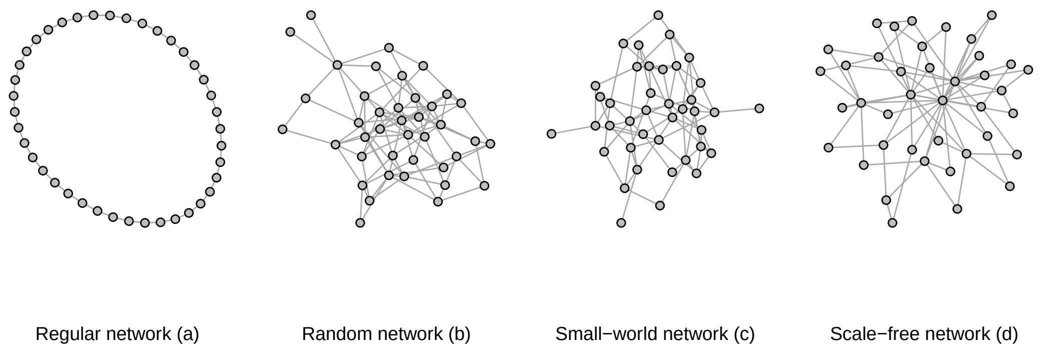


Figure 3: Example of networks with different topologies.

random connections. In contrast, the scale-free model (Barabási & Albert, 1999), another well-known complex network model, introduces connectivity distributions obeying a power law. In this case, network connections evolve according to an exponential function γ of preferential attachments.

Because the Kuramoto model can handle globally connected networks, it is necessary to consider an adaptation of the original model in order to ensure synchronization ability in complex networks as well (Arenas, Daz-Guilera, Kurths, Moreno, & Zhou, 2008), according to Equation 5:

$$\frac{d\theta_i}{d_t} = \omega_i + \sum_{j=1}^{N} a_{ij} \sin(\theta_j - \theta_i), \tag{5}$$

where a_{ij} represents the coupling strength $\forall i \neq j$. For each pair of noncoupled oscillators i and j, the coupling strength is null; that is, $a_{ij} = 0$.

5. COOPT: Coupled Oscillator OPTimization Algorithm

In general, a typical DCOP formulation displays similar properties to those found in a coupled oscillator network. In a DCOP, agents aim to coordinate their actions in order to optimize the global objective function defined as a set of constraints (Lesser et al., 2003). However, synchronization is related to frequency convergence and phase coherence in a coupled oscillator network (Strogatz, 2003). Therefore, both formulations seek to reach a consensus in a group of interacting units.

Mathematical models of synchronization in coupled oscillators presented in Section 4 have been demonstrated their potential to coordinate individual actions on complex, large networks. Inspired by collective behavior of coupled oscillators, we propose COOPT, a new anytime DCOP algorithm that introduces concepts of synchronization dynamics for speeding up the convergence process towards high-quality solutions.

5.1 Definitions

At the aim of providing a new anytime local search algorithm for large-scale DCOPs, firstly we introduce a mapping between concepts of DCOP and coupling oscillators network. For convenience, we assume that each agent a_i holds a single variable x_i and we use the terms agent and variable interchangeably.

5.1.1 Variables as Oscillators

An intuitive analogy between a typical DCOP formulation and synchronization in coupled oscillator network consists of representing variables as oscillators and constraints as coupling strengths between oscillators, where an agent a_i has an individual behavior similar to an oscillator i. Values $d \in D_i$ represent the domain for the variable x_i, where d_i refers to the current state of x_i.

Definition 1. *Agent: each agent a_i holds a local view which represents a partial set of assignments of the form $\{(x_j, d_j), ...\}$. The local view contains the current assignments from neighboring variables $x_j \in N_i$, where N_i represents a set of variables that shares constraints with x_i.*

Definition 2. *Local cost: the local cost of a given partial assignment for the agent a_i is defined by $\delta(d_i) = \sum\limits_{x_j \in view} f_{ij}(d_i, d_j)$, when $x_i \leftarrow d_i$ and $x_j \leftarrow d_j$ according to a_i's view. On the other hand, δ represents the global cost of a complete solution and δ^* is the best global cost in a given problem.*

In contrast, the Kuramoto model describes the influence of local interactions from the frequency ω_i of each oscillator. Therefore, the local cost $\delta(d_i)$ is analogous to the frequency ω_i, making each oscillator updates its individual frequency from the current states of the neighboring oscillators.

Definition 3. *Frequency: the local costs $\delta(d)$ for each $d \in D_i$, considering all assignment combinations of the neighboring variables $x_j \in N_i$, represent the possible discrete frequencies ω_i for a given variable x_i.*

Nevertheless, a locally optimal solution at a given agent can result in conflicts of interest in other agents due to the individual objectives and partial views on the problem, thus possibly producing worse global solutions. The Kuramoto Model captures such conflicts of interest from the phases difference between neighboring oscillators, as described by the order parameter $r(t)$ which measures the level of coherence of the system (Equation 4). Therefore, reaching phase coherence ensures a global observation from individual actions and partial solutions.

Definition 4. *Phase: the phase θ_i represents the level of coherence between a variable assignment and a given partial solution; that is, an error degree between the local cost $\delta(d_i)$ of x_i and its neighboring variables $x_j \in N_i$. Thus, such observation represents a level of global coherence (consensus) measured from the phases of all the agents at a given moment.*

5.1.2 Cost Functions as Couplings

Classical synchronization models maintain a static coupling strength among oscillators. Nevertheless, some studies (Skardal & Restrepo, 2012; Meier, Haschke, & Ritter, 2014; Gushchin, Mallada, & Tang, 2015) have addressed variations of the Kuramoto model by introducing coupling strength as symmetric functions. Such models represent couplings between pairs of oscillators in which coupling strengths can vary according to the current state of the oscillators.

Such synchronization models are more appropriate for representing symmetric constraint functions in a DCOP formulation. As a consequence, the static coupling strength a_{ij} needs to be replaced by a function f_{ij}, according to Equation 6:

$$\frac{d\theta_i}{d_t} = \omega_i + \frac{K}{N}\sum_{j=1}^{N} f_{ij}(x_i, x_j)\sin(\theta_j - \theta_i), \tag{6}$$

where x_i and x_j represent the current states of each pair of coupled oscillators i and j.

5.1.3 Synchronization Dynamics

Updating the frequencies ω_i is a key step of synchronization dynamics in an oscillator network. The synchronization drives the oscillators toward a frequency mean-field Ω by redefining iteratively their phases. Such iterative process results in a reduction of the phases deviation among the oscillators. In a DCOP formulation, phase represent an error degree between a variable assignment and its neighboring variables.

Therefore, the synchronization dynamics in a DCOP must lead the agents to a coherence of their actions, even if there are conflicting local actions. In this sense, some studies on clustering detection in oscillator networks (Arenas, Daz-Guilera, & Prez-Vicente, 2006; Hong & Strogatz, 2011; Wu, Jiao, Li, & Chen, 2011; Meier et al., 2014) are candidates for addressing such consensus problems, which naturally require coordination of the agents for group formation.

Arenas et al. (2006) demonstrated that mean-field-based synchronization models are inefficient for identifying the effects of the local dynamics of the oscillators. The authors introduced a local order parameter for measuring the correlation between pairs of oscillators instead of only considering a global observation, according to Equation 7:

$$\rho_{ij}(t) = \langle \cos(\theta_j - \theta_i) \rangle, \tag{7}$$

where the cosine of the phases difference is a similarity measure for detecting functional groups or communities of oscillators at a given t time.

Meier et al. (2014) introduced an alternative synchronization model for perceptual grouping in coupled oscillator networks. This model uses discrete frequencies $\omega_\alpha = \alpha\omega_0$ from a discrete and finite $\mathcal{N}(\alpha) = \{1, ..., L\}$ set of possible α states arranged in a L-layered architecture, where $\alpha \in \mathcal{N}(\alpha)$ and ω_0 is the initial frequency. The frequencies iteratively evolve following Equations 8 and 9:

$$S_i(\alpha) = \sum_{i \in \mathcal{N}(\alpha)} f_{ij}\frac{1}{2}(\cos(\theta_j - \theta_i) + 1) \tag{8}$$

$$\omega_i = \omega_0 \ \operatorname{argmax}(S_i(\alpha)), \tag{9}$$

where every frequency ω_i is updated using a cosine-based similarity measure for detecting groups of oscillators with similar features.

Our DCOP formulation is inspired by aforementioned clustering detection techniques in coupled oscillator networks. This formulation introduces an optimization mechanism that seeks to reduce both the local costs and the error degree between individual actions in groups of agents. So the synchronization dynamics in our formulation evolve according to Equations 10, 11, and 12:

$$S_i = \delta(d) + \sum_{x_j \in view} f_{ij}(d, d_j)\frac{1}{2}(\rho_{ij} + 1) \qquad \forall d \in D_i \tag{10}$$

$$d_i = \text{argmin}(S_i) \tag{11}$$

$$\omega_i = \delta(d_i), \tag{12}$$

where the factor $\frac{1}{2}$ maintains the similarity measure within the interval $[0, 1]$. When the similarity measure $\rho_{ij} \approx 0$, x_j does not present a tendency of conflict on the partial solution and the synchronization process of x_i considers exclusively the cost of f_{ij}. Otherwise, $\rho_{ij} \approx 1$ indicates a trend of conflict of x_j in the partial solution and the synchronization process of x_i considers an additional factor on the cost of f_{ij}.

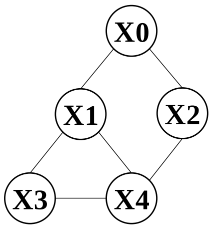
Definition 5. *Synchronization process: S_i represents a vector containing the sum of the local costs and the phase similarities from local view of x_i for each $d \in D_i$ (Equation 10). Then a new value d_i is chosen from the smallest value in S_i (Equation 11), where ω_i refers to the local cost $\delta(d_i)$ when $x_i \leftarrow d_i$ (Equation 12).*

5.2 Communication Model

The communication model of COOPT requires a partial ordering among the agents. Such ordering uses a spanning-tree structure in which all agents have a parent route to the root agent and all parents are neighbors of their children in the constraint graph. The purpose of generating a spanning-tree arrangement from the constraint graph is to provide a search process that increases monotonically the global solution quality over time. To achieve this goal, all agents must be able to know the best solution found so far during the search.

In a spanning-tree arrangement for a DCOP, each agent can have a single parent p_i and multiple children $x_j \in C_i$, where $p_i = \varnothing$ indicates that x_i represents the root agent and $C_i = \emptyset$ means that x_i is a leaf of the spanning-tree. In COOPT, this ordering allows the root agent to calculate the aggregated cost of a complete solution in a given moment. A simple way to obtain a spanning-tree ordering consists of performing a breadth-first search exploration from the constraint graph.

The algorithm works using three types of messages: *assign*, *cost*, and *synchronize*, which determine the internal stage (either synchronization or cost propagation) of the agents during the search. The *assign* messages aim to notify neighbors $x_j \in N_i$ about new values chosen by x_i. The *cost* messages inform higher priority agent p_i about the aggregated cost of the sub-tree rooted at x_i during the cost propagation stage. The *synchronize* messages inform lower priority agents $x_j \in C_i$ they must perform a new assignment during the synchronization stage.



f_{ij}	d_i	d_j
2	0	0
1	0	1
1	1	0
3	1	1

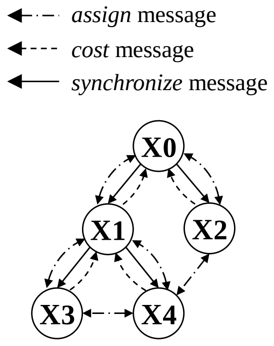


Figure 4: Example of DCOP and COOPT's communication model.

Figure 4 presents the communication model of COOPT from an example of DCOP involving five variables with a binary domain and six constraints sharing a single cost function. COOPT implements an anytime local search by an iterative search process, where each iteration is composed by a communication round. A communication round is defined as involving multiple broadcasts of messages between neighboring agents. Each agent has a *counter* which increases by one after performing each synchronization stage. So the agents terminate the search process when the n-th synchronization stage is performed.

5.3 Algorithm Description

Algorithm 1 shows the procedures of COOPT. According to definitions introduced in Section 5.1, each agent has a local *view* representing a partial solution of the form $\{(x_j, [d_j, \theta_j]), ...\}$ that, in addition to the assignments of neighboring variables, also holds their phases in some round. Each agent also stores information about the best round, assignment, and aggregated cost in $bestRound$, $bestValue$ and $bestCost$, respectively. These information describe the best global solution found so far during the search.

The search starts concurrently by the Initialize procedure. Each agent x_i assigns a value that exclusively minimizes its local cost $\delta(d)$ and initializes its phase θ_i with a random value following a Gaussian distribution (lines 2-3). Then *assign* messages are sent from x_i to its neighbors informing them about the new values of d_i and θ_i (lines 8-10). At the end of this procedure, if x_i is a leaf of the spanning-tree, a *cost* message is sent to the parent informing it about the local cost of x_i (lines 11-14).

Agents update their local *view* by the Assign procedure whenever new *assign* messages are received (line 17). After an agent updates its local *view* with information collected from some neighbor, the ComputeCost or PerformSynch procedures can be performed according to the internal stage of the agent (lines 18-23). The ComputeCost procedure aim to propagate the aggregated cost of the sub-tree rooted at x_i. The PerformSynch procedure performs a new synchronization stage.

The Cost procedure is performed after receiving a *cost* message. Subsequently, x_i stores the aggregated cost of a partial solution (line 26). When x_i receives the aggregated costs of all children (line 27) and its local *view* is up-to-date with information collected

Algorithm 1 Procedures of COOPT

```
1:  function INITIALIZE
2:      $\theta_i \sim \mathcal{N}(\mu, \sigma^2)$
3:      $d_i, bestValue \leftarrow d \in D_i \mid \min(\delta(d))$
4:      $counter, bestRound \leftarrow 1$
5:      $bestCost \leftarrow \infty$
6:      $view, costs \leftarrow \emptyset$
7:      $cost, sync \leftarrow$ false
8:      for each $x_j \in N_i$ do
9:          Send ASSIGN$(x_i, d_i, \theta_i)$ to $x_j$
10:     end for
11:     if $C_i = \emptyset$ then
12:         $cost \leftarrow$ true
13:         COMPUTECOST
14:     end if
15: end function
16: function ASSIGN$(x_j, d_j, \theta_j)$          ▷ upon receiving assign
17:     $view \leftarrow view \cup \{(x_j, d_j, \theta_j)\}$
18:     if $cost$ then
19:         COMPUTECOST
20:     end if
21:     if $sync$ then
22:         PERFORMSYNCH
23:     end if
24: end function
25: function COST$(x_j, \delta_j)$          ▷ upon receiving cost
26:     $costs \leftarrow costs \cup \{(x_j, \delta_j)\}$
27:     if $\forall x_j \in C_i \mid \exists x_j \in costs$ then
28:         $cost \leftarrow$ true
29:         COMPUTECOST
30:     end if
31: end function
32: function SYNCH$(t)$          ▷ upon receiving synchronize
33:     $sync \leftarrow$ true
34:     if $t \neq bestRound$ then
35:         $bestRound \leftarrow t$
36:         $bestValue \leftarrow d_i$
37:     end if
38:     PERFORMSYNCH
39: end function
40: function PERFORMSYNCH
41:     if $\forall x_j \in N_i \mid \exists x_j \in view$ then
42:         $sync \leftarrow$ false
43:         $counter \leftarrow counter + 1$
44:         if $counter < n$ then
45:             $S_i \leftarrow \emptyset$
46:             for each $d \in D_i$ do
47:                 $\omega \leftarrow \delta(d) + \sum_{x_j \in N_i} f_{ij}(d_i, d_j)\frac{1}{2}(\rho_{ij} + 1)$
48:                 $S_i \leftarrow S_i \cup \{(d, \omega)\}$
49:             end for
50:             $d_i \leftarrow \underset{\omega \in S_i}{\text{argmin}}(S_i)$
51:             $\omega_i \leftarrow \delta(d_i)$
52:             $\theta_i \leftarrow \omega_i + \frac{K}{N} \sum_{x_j \in N_i} f_{ij}(d_i, d_j) \sin(\theta_j - \theta_i)$
53:             for each $x_j \in N_i$ do
54:                 Send ASSIGN$(x_i, d_i, \theta_i)$ to $x_j$
55:             end for
56:         end if
57:         if $C_i = \emptyset$ then
58:             if $counter < n$ then
59:                 COMPUTECOST
60:             end if
61:         else
62:             for each $x_j \in C_i$ do
63:                 Send SYNCH$(bestRound)$ to $x_j$
64:             end for
65:         end if
66:     end if
67: end function
68: function COMPUTECOST
69:     if $\forall x_j \in N_i \mid \exists x_j \in view$ then
70:         $cost \leftarrow$ false
71:         $\delta_i \leftarrow \delta(d_i) + \sum_{x_j \in costs} \delta_j$
72:         if $P_i = \varnothing$ then
73:             if $bestCost > \delta_i$ then
74:                 $bestValue \leftarrow d_i$
75:                 $bestCost \leftarrow \delta_i$
76:                 $bestRound \leftarrow counter$
77:             end if
78:             PERFORMSYNCH
79:         else
80:             Send COST$(x_i, \delta_i)$ to $P_i$
81:         end if
82:     end if
83: end function
```

of all its neighbors in some round (line 69), the aggregated cost of the sub-tree rooted at x_i can be computed by the ComputeCost procedure (lines 71). If x_i is the root of the spanning-tree, this cost represents the global cost δ of a complete solution. So x_i stores the best cost, round and assignment (lines 73-77) and performs a new synchronization stage by PerformSynch procedure (line 78).

The Synch procedure is performed whenever an agent receives a *synchronize* message. Thus, x_i updates its best round and assignment found so far (lines 34-37). If x_i's local *view* is up-to-date with information collected of all its neighbors in some round (line 41), a new synchronization stage can be performed. Upon starting the PerformSynch procedure, x_i increments its synchronization stage *counter* (line 43). If x_i does not reach the n-th synchronization stage (line 44), x_i calculates ω for each value $d \in D_i$ and inserts it into S_i (lines 46-49). Next, x_i chooses a new value d_i that reduces $\omega \in S_i$ and updates ω_i with the new local cost $\delta(d_i)$, and θ_i from the new value of ω_i (lines 50-52).

Also at the PerformSynch procedure, if x_i is a leaf of the spanning-tree and the n-th synchronization stage does not reach (i.e., no termination condition is met), x_i performs ComputeCost procedure (line 57-60). Otherwise, if x_i is not a leaf of the spanning-tree, x_i sends *synchronize* messages to its children containing information about the best round at the moment (lines 62-64).

5.4 Anytime Property

COOPT implements the *Anytime Local Search DCOP* framework (ALS_DCOP) designed for local search algorithms (Zivan et al., 2014). In ALS_DCOP, agents perform variable assignments using some local search method. Next the aggregated costs of each sub-tree rooted at a given agent are propagated from the leaves towards the root of the spanning-tree. At the end of each communication round, the root agent computes the aggregated cost of the current complete solution and sends this information for all agents from the root towards the leaves of the spanning-tree.

Lemma 1. *For a given communication round t, all agents hold a up-to-date local view about the neighboring assignments and the root of the spanning-tree is able to compute the aggregated cost of the complete solution at the t round.*

Proof. We prove by induction from the recursive definition of the sub-tree rooted at x_i. If x_i is a leaf of the spanning-tree ($C_i = \emptyset$), the aggregated cost of the sub-tree rooted at x_i is equal to the local cost $\delta(d_i)$. If x_i is not a leaf of the spanning-tree ($C_i \neq \emptyset$), the aggregated cost of the sub-tree rooted at x_i is equal to the local cost $\delta(d_i)$ plus the aggregated cost reported from all $x_j \in C_i$ at the t round. Therefore, if x_i is the root of the spanning-tree ($p_i = \varnothing$), x_i is able to compute the aggregated cost of the complete solution at the t round. □

Lemma 2. *All agents hold the best round and the best assignment found until the t-th round. When the search is finished, all agents know the assignments of their variables that produced the best global cost so far.*

Proof. According to the Lemma 1, the root of the spanning-tree ($p_i = \varnothing$) can compute the aggregated cost for each round. Thus, the root agent holds the index of the round that has produced the best global cost. When finishing the t-th round, the root agent compares the aggregated cost of the t round with the best round known in $t-1$. Then the root agent notifies all children ($x_j \in C_i$) until reaching leaves of the spanning-tree about the best round explored so far. $\square$

Theorem 1. *COOPT is anytime from a monotonic local search that results in the best complete solution found until the t-th round.*

Proof. This is a direct consequence of Lemmas 1 and 2. At the end of each round, all agents know their assignments that have resulted in the best complete solution so far. Therefore, COOPT returns the best complete solution found within a predefined rounds limit. $\square$

5.5 Example Run

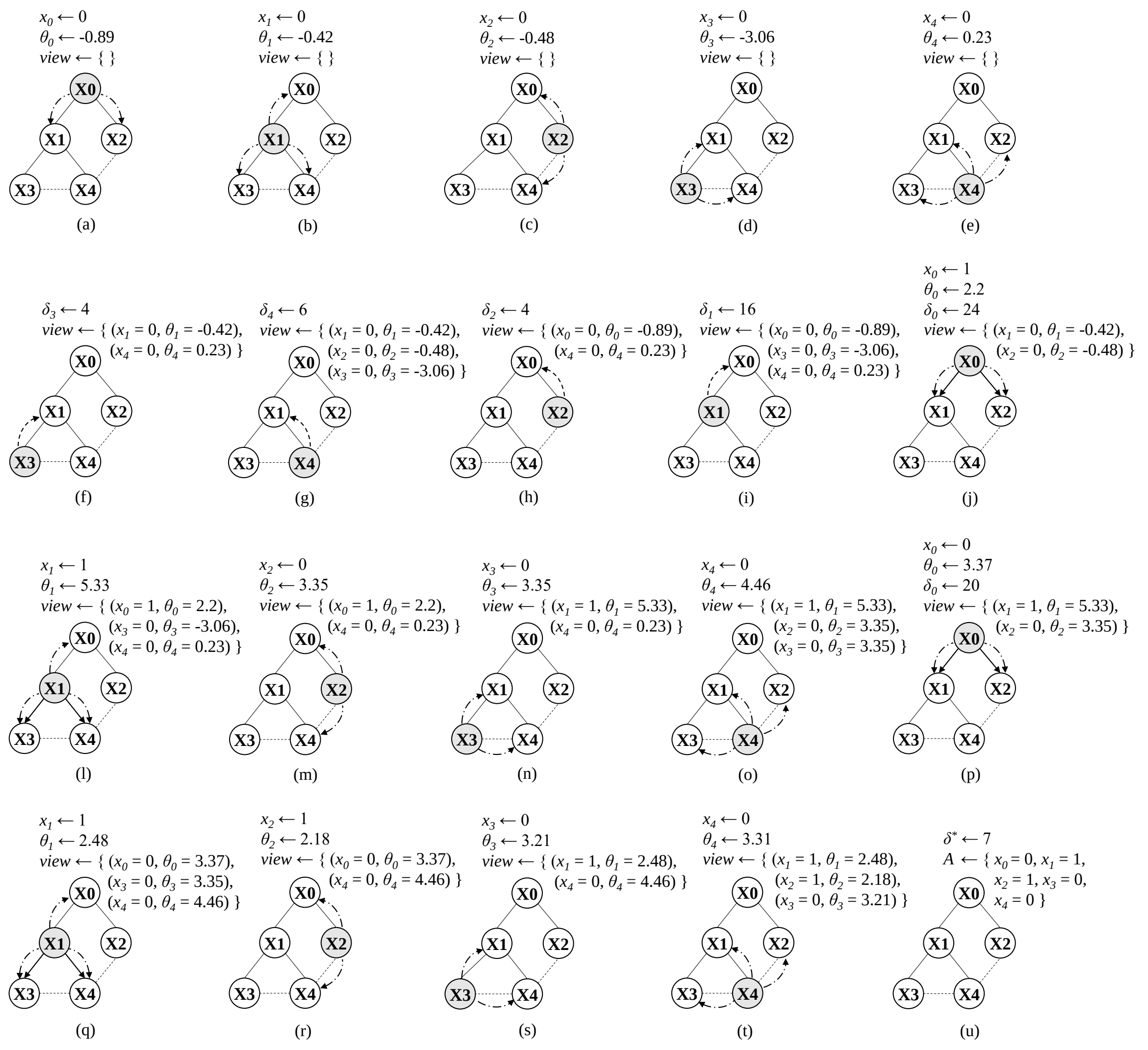
Figure 5 illustrates part of COOPT execution from the example of DCOP presented in Figure 4. Because in COOPT the actions of all agents in a single round can be performed concurrently; that is, a synchronous behavior, we demonstrate one possible sequence of execution considering $n = 2$ synchronization stages for each agent. This demonstration took into account a global coupling strength $K = 1$.

First, all agents concurrently choose values $\forall d \in D_i$ that exclusively minimizes their local costs $\delta(d_i)$, initialize their phases randomly $\theta_i \sim \mathcal{N}(\mu, \sigma^2)$ by a Gaussian distribution function with mean $\mu = 0$ and standard deviation $\sigma^2 = 1$ and create empty local *views*. Then all agents send *assign* messages to their neighbors informing them about the new values of d_i and θ_i (Figures 5.a, 5.b, 5.c, 5.d, and 5.e).

Next, x_3 updates its local *view* from *assign* messages sent to it in the previous round. After updating its local *view*, x_3 calculates the aggregated cost $\delta_3 = 4$ and sends a *cost* message to x_1 containing such information (Figure 5.f). Since x_3 is a leaf of the spanning-tree, the aggregated cost δ_3 is equal to its local cost $\delta(d_i)$. The same procedures of cost propagation stage are performed by x_4 and x_2 because they are also leaves, therefore, they must report to their parents the aggregated costs $\delta_4 = 6$ and $\delta_2 = 4$ respectively by *cost* messages (Figures 5.g and 5.h).

After x_1 has updated its local *view* with neighboring assignments from *assign* messages and collected the aggregated cost of its children from *cost* messages sent to it in the previous rounds, x_1 can perform procedures of cost propagation stage. Thus, x_1 calculates the aggregated cost $\delta_1 = 16$ of the sub-tree rooted at itself and notifies x_1 about this information by a *cost* message (Figure 5.i). Because x_1 is not a leaf of the spanning-tree, the aggregated cost δ_1 takes into account its local cost $\delta(d_i)$ plus the aggregated costs reported by its children; that is, $\delta_3 = 4$ and $\delta_4 = 6$.

Similar to x_1, x_0 performs the procedures of cost propagation stage after receiving all *assign* and *cost* messages sent to it in the previous rounds. However, x_0 is the root of the spanning-tree; that is, its aggregated cost $\delta_0 = 24$ represents the global cost of a complete solution. Next, x_0 starts a synchronization stage by choosing a new value $x_0 \leftarrow 1$ and updating its phase $\theta_0 \leftarrow 2.2$. Then x_0 sends *assign* messages to its neighbors informing them about new values and sends *synchronize* messages to its children start a synchronization stage (Figure 5.k).



x0 ← 0, θ0 ← -0.89, view ← { }
(a)
x1 ← 0, θ1 ← -0.42, view ← { }
(b)
x2 ← 0, θ2 ← -0.48, view ← { }
(c)
x3 ← 0, θ3 ← -3.06, view ← { }
(d)
x4 ← 0, θ4 ← 0.23, view ← { }
(e)
δ3 ← 4, view ← { (x1 = 0, θ1 = -0.42), (x4 = 0, θ4 = 0.23) }
(f)
δ4 ← 6, view ← { (x1 = 0, θ1 = -0.42), (x2 = 0, θ2 = -0.48), (x3 = 0, θ3 = -3.06) }
(g)
δ2 ← 4, view ← { (x0 = 0, θ0 = -0.89), (x4 = 0, θ4 = 0.23) }
(h)
δ1 ← 16, view ← { (x0 = 0, θ0 = -0.89), (x3 = 0, θ3 = -3.06), (x4 = 0, θ4 = 0.23) }
(i)
x0 ← 1, θ0 ← 2.2, δ0 ← 24, view ← { (x1 = 0, θ1 = -0.42), (x2 = 0, θ2 = -0.48) }
(j)
x1 ← 1, θ1 ← 5.33, view ← { (x0 = 1, θ0 = 2.2), (x3 = 0, θ3 = -3.06), (x4 = 0, θ4 = 0.23) }
(l)
x2 ← 0, θ2 ← 3.35, view ← { (x0 = 1, θ0 = 2.2), (x4 = 0, θ4 = 0.23) }
(m)
x3 ← 0, θ3 ← 3.35, view ← { (x1 = 1, θ1 = 5.33), (x4 = 0, θ4 = 0.23) }
(n)
x4 ← 0, θ4 ← 4.46, view ← { (x1 = 1, θ1 = 5.33), (x2 = 0, θ2 = 3.35), (x3 = 0, θ3 = 3.35) }
(o)
x0 ← 0, θ0 ← 3.37, δ0 ← 20, view ← { (x1 = 1, θ1 = 5.33), (x2 = 0, θ2 = 3.35) }
(p)
x1 ← 1, θ1 ← 2.48, view ← { (x0 = 0, θ0 = 3.37), (x3 = 0, θ3 = 3.35), (x4 = 0, θ4 = 4.46) }
(q)
x2 ← 1, θ2 ← 2.18, view ← { (x0 = 0, θ0 = 3.37), (x4 = 0, θ4 = 4.46) }
(r)
x3 ← 0, θ3 ← 3.21, view ← { (x1 = 1, θ1 = 2.48), (x4 = 0, θ4 = 4.46) }
(s)
x4 ← 0, θ4 ← 3.31, view ← { (x1 = 1, θ1 = 2.48), (x2 = 1, θ2 = 2.18), (x3 = 0, θ3 = 3.21) }
(t)
δ* ← 7, A ← { x0 = 0, x1 = 1, x2 = 1, x3 = 0, x4 = 0 }
(u)
X0
X1
X2
X3
X4

Figure 5: Example of COOPT execution from the DCOP showed in Figure 4.

After x_1 and x_2 have received all *assign* and *synchronize* messages sent to them in the previous rounds, they perform the synchronization stage by choosing new values $x_1 \leftarrow 1$ and $x_2 \leftarrow 0$ and updating their phases $\theta_1 \leftarrow 5.33$ and $\theta_2 \leftarrow 3.35$. Then x_1 and x_2 send *assign* messages to their neighbors, but in contrast to x_2, x_1 sends *synchronize* messages its children start a synchronization stage (Figures 5.l and 5.m).

So x_3 and x_4 can start a synchronization stage after receiving *assign* messages from neighbors and *synchronize* message from x_1. Then x_3 and x_4 choose new values $x_3 \leftarrow 0$ and $x_4 \leftarrow 0$, update their phases $\theta_3 \leftarrow 3.35$ and $\theta_4 \leftarrow 4.46$, and send *assign* messages to their neighbors (Figures 5.n and 5.o). At this moment, agents are able to propagate the aggregated costs from the leaves towards the root of the spanning-tree, so that the root calculates the global cost for the new complete solution. These intermediate steps for calculating the global cost were omitted for readability because are procedures of cost propagation stage have already been discussed.

After x_0 has calculated the aggregated cost $\delta_0 = 20$, the last synchronization stage is started. So x_0 chooses a new value $x_0 \leftarrow 0$, updates its phase $\theta_0 \leftarrow 3.37$ and sends *assign* messages to its neighbors and *synchronize* messages to its children (Figure 5.p). The other variables also perform the last synchronization stage by choosing new values $x_1 \leftarrow 0$, $x_2 \leftarrow 1$, $x_3 \leftarrow 0$, and $x_4 \leftarrow 0$, updating their phases $\theta_1 \leftarrow 2.48$, $\theta_2 \leftarrow 2.18$, $\theta_3 \leftarrow 3.21$, and $\theta_4 \leftarrow 3.31$, and sending *assign* messages to their neighbors and *synchronize* messages to their children (Figures 5.q, 5.r, 5.s, and 5.t).

Finally, after all agents have performed procedures of cost propagation stage, x_0 is able to compute the aggregated cost $\delta_0 = 14$ of the new complete solution. Thus, one of the best possible solution was found, where $A = \{x_0 \leftarrow 0; x_1 \leftarrow 1; x_2 \leftarrow 1; x_3 \leftarrow 0; x_4 \leftarrow 0\}$ with a optimum global cost $\delta^* = 7$ (Figure 5.u).

5.6 Phase Coherence

The emergent collective behavior of COOPT leads to the formation of groups of agents, aiming at coherence among individual actions. In other words, the trend of synchronization of COOPT causes agents to be attracted by groups so as to minimize the error degree in a given partial solution. Therefore, the collective behavior tries to converge gradually the system towards phase coherence.

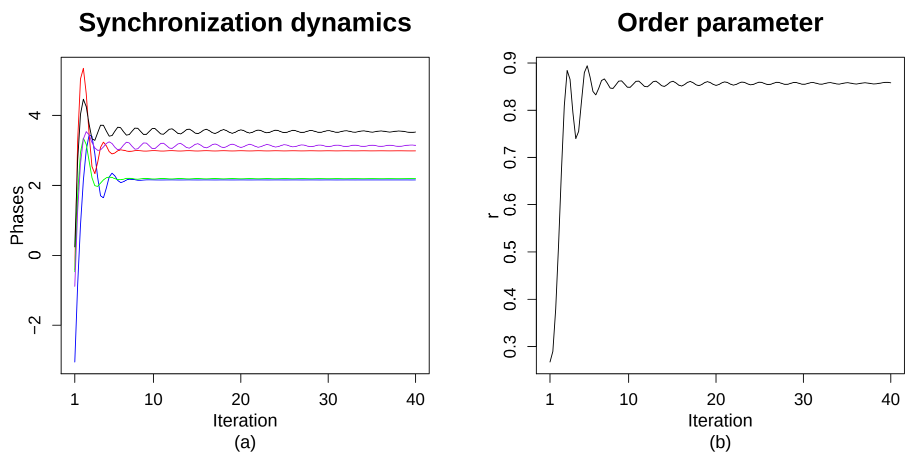


Figure 6: Trend of synchronization of COOPT from the DCOP showed in Figure 4.

Figure 6 illustrates the trend of synchronization and phase coherence in a execution of COOPT from the example of DCOP discussed in Section 5.5. In contrast, this running example has considered a limit of $n = 40$ synchronization stages per agent. Figure 6.a depicts the synchronization dynamics and gradual phase adjustments over time. In addition to synchronization dynamics, Figure 6.b shows the phase coherence over time using the order-parameter r described by Equation 4.

5.7 Complexity Analysis

We provide the complexity analysis taking into account actions performed sequentially by agents. For each iteration of the algorithm; that is, a single execution of synchronization and cost propagation stage, agents run communication rounds in which each agent exchanges *assign*, *cost*, and *synchronize* messages between neighbors. The number of *synchronize* and *cost* messages sent is $|\mathcal{V}| - 1$. In addition, *assign* messages are sent $2\,|\mathcal{E}|$ times. Thus, the number of exchanged messages is linear according to $\mathcal{O}(2\,|\mathcal{E}| + 2(|\mathcal{V}| - 1))$.

The *assign* message contains the assignment and phase of a neighbor variable in some round. In contrast, the *synchronize* message propagates the index of the best round found so far. Finally, the *cost* message contains the aggregated cost of a partial solution from the sub-tree rooted at some child variable. Therefore, all messages have a constant size. Regarding the local computation, the number of constraint checks is linear according to $\mathcal{O}(2\,|\mathcal{E}|\max(|D|))$, where $\max(|D|)$ represents the largest domain $D \in \mathcal{D}$.

In terms of space, each agent holds the assignments and phases of its neighbors and the aggregated cost of its children reported in previous round. Agents also hold information about the best round and the assignment found so far. This results in a $\mathcal{O}(\max(|N|))$ linear space, where $\max(|N|)$ represents the largest neighboring among all variables. Thus, COOPT is a low-overhead anytime algorithm, because agents exchange a linear number of messages and perform a linear number of operations.

With respect to the concurrent execution of COOPT, the overhead is determined by the height of the spanning-tree. In other words, each path from the root to the leaves contains individual actions that are performed sequentially. COOPT benefits when the spanning-tree has a minimum height because of the independent paths that can be solved concurrently. So we suggest generating the spanning-tree ordering from a breadth-first search because this search tends to find trees that have only logarithmic height. Since a breadth-first search results in a shortest path from the root to leaves, the height of the spanning-tree is expected to be small even on dense constraint graphs (Zivan et al., 2014).

6. Experimental Evaluation

In order to evaluate the performance of the proposed algorithm on large instances of DCOP, our experiments considered two types of problems: random problems and realistic meeting scheduling problems. Random problems have wide generality and are useful to reflect the real ability of solving DCOP. In contrast, meeting scheduling problems (MSP) capture fundamental characteristics of real-world problems involving joint activities. Experiments were performed in FRODO[1] framework on a machine with an Intel(R) Core(TM) i7-7500U CPU with 2.9 Ghz and 16GB of RAM.

6.1 Random Problems

Random problems represent minimization decision problems (Leite et al., 2014) in our experiments, in which each agent $a_i \in \mathcal{A}$ holds a single variable $x_i \in \mathcal{X}$. We considered medium and large random problems involving 50, 100, and 200 variables with a single

1. FRODO is a popular framework for solving DCOP (Léauté, Ottens, & Szymanek, 2009)

domain size $|D| = 10$. Constraints were binaries cost functions $f_{ij} \in \mathcal{R}$ and costs were randomly chosen within the interval $[0, 100]$.

The process of generating our random problems explored uniform and non-uniform constraint networks by introducing different connection patterns. Several studies in MAS research field have shown that the network topology has a significant impact on distributed problem solving (Gaston & DesJardins, 2005; Le, Son, Pontelli, & Yeoh, 2015; Le, Fioretto, Yeoh, Son, & Pontelli, 2016). Therefore, our random problems covered different levels of complexity in terms of three dimensions: i) number of variables; ii) constraint density; and iii) network topology.

Therefore, our random problems considered graph constraints with densities from the range $p \in \{0.3, 0.5, 0.7, 0.9\}$. With respect to the network topologies, we considered the following models: regular (Wasserman & Faust, 1994), random (ER) (Erdös & Rényi, 1959), small-world (WS) (Watts & Strogatz, 1998), and scale-free (BA) (Barabási & Albert, 1999). In addition, we set the parameters $c = 0.2$ for rewiring connections and $\gamma = 1$ for linear preferential attachment in small-world and scale-free models, respectively. We generated 30 instances of random problems for each combination of variables, densities, and topologies.

6.2 Meeting Scheduling Problems

Meeting scheduling problems refer to agents that are trying to schedule events in certain time slots (Modi & Veloso, 2004). Scheduling conflicts occurs if two or more events are scheduled in overlapping time slots. A meeting scheduling problem in our experiments involved a set $\mathcal{A}$ of agents, a number M of meetings with N agents intended to attend it per meeting, and a number T of time slots.

Our experiments involving meeting scheduling problems used a DCOP formulation known as private events as variables (PEAV) (Maheswaran et al., 2004b). In PEAV, agents make decisions only about the events in which they participate. Such formulation describes a variable $x_n(t) \in \mathcal{X}$ representing the n-th event for a given agent at the t-th time slot. Thus, we had $M \times N$ variables per instance.

The setup for meeting scheduling problems in our experiments included: i) 220 agents, 80 meetings, and 5 agents per meeting; ii) 220 agents, 40 meetings, and 10 agents per meeting; and iii) 220 agents, 20 meetings, and 20 agents per meeting. All setups had 20 available time-slots for each meeting. Scheduling conflicts incurred costs equals to 1000 and each agent also has preferences in each of the available time slots, in which costs were randomly chosen within the interval $[0, 100]$. We generated 30 instances following these setups, where each DCOP was composed by 220 agents and 400 variables with a single domain size $|D| = 20$.

6.3 Evaluation Method

Our experimental evaluation benchmarks COOPT with state-of-the-art incomplete DCOP algorithms, named DSA (Fitzpatrick & Meertens, 2003), DSA-SDP (Zivan et al., 2014), DUCT (Ottens et al., 2017), GDBA (Okamoto et al., 2016), MGM (Zhang et al., 2005), and MGM-2 (Pearce et al., 2008). Unfortunately we could not compare DUCT on large problems involving 200 or more variables due to the computational effort needed. We also assume $n = 100$ iterations for all local search algorithms.

The results were averaged over 30 executions for each algorithm and problem instance. We perform a performance evaluation using well-known computation and communication metrics, such as number of messages, amount of exchanged information (Modi et al., 2005), and non-concurrent constraint checks (NCCC) (Meisels, Kaplansky, Razgon, & Zivan, 2002). Because all evaluated algorithms are incomplete methods, our performance analysis also compares solution costs.

We applied the Friedman and Nemenyi nonparametric statistical tests for comparing the performance of the algorithms from the average results. The Friedman test allows detection of whether there is a statistical significance between the results, even on nonparametric conditions (Friedman, 1940). Under the null hypothesis H_0, Friedman test states that all the algorithms have equivalent performance, whereas a rejection implies the existence of differences among them.

The Friedman test begins by an ordinal ranking of the results for each metric. The test is calculated by $Q = \frac{12}{mk(k+1)} \sum_{j=1}^{k} R_j^2 - 3m(k+1)$, where m is the total number of instances, k is the number of degrees of freedom, and R_j^2 is the square of the rank total for each algorithm. The test rejects H_0 depending on whether Q is greater than a given α significance level. After this test, a post-hoc test, such as Nemenyi, can be used for providing a performance ranking of the evaluated algorithms.

Therefore, the Nemenyi post-hoc test produces an ascending ranking by pair-wise tests (Nemenyi, 1963). This ranking is generated by a critical distance ($|\overline{R}_i - \overline{R}_j|$) for each rank pair on a given level of significance. The critical distance for the Nemenyi test is calculated by $CD = \frac{q_\alpha}{\sqrt{2}}\sqrt{\frac{k(k+1)}{6m}}$, where m is the total number of instances, k is the number of degrees of freedom, and q_α represents the confidence level.

6.4 Algorithms and Parameters

All algorithms, except MGM and MGM-2, need some particular parameters during the search, so we provide details on the selected values for such parameters in the experiments. DSA uses an activation probability p before choosing new assignments, and we considered $p = 0.7$ as reported by Zhang et al. (2005). We considered DSA version C because its decision process is known to be more aggressive than other versions (Zhang et al., 2005), being more appropriate for general valued DCOPs. DSA-SDP has four parameters p_A, p_B, p_C, and p_D which represent the potential of improvement for calculating the probability of choosing new assignments, so we use values $p_A = 0.6$, $p_B = 0.15$, $p_C = 0.4$, and $p_D = 0.8$ as suggested by authors (Zivan et al., 2014).

Regarding the DUCT, its termination condition occurs when the difference of the best value and the minimal lower bound is smaller than the ϵ confidence bound, so we assumed $\epsilon = 0.05$ according to the authors (Ottens et al., 2017). Finally, GDBA provides three different types of behavior for replacing variable assignments. Therefore, we setup GDBA with multiplicative weight manner (M), non-minimum constraint violation (NM), and table scope of cost increase (T), resulting in (M, NM, T) combination for assignment replacement according to the authors (Okamoto et al., 2016).

On the other hand, COOPT requires a parameter K that represents the global coupling strength among connected variables. Nevertheless, the global coupling strength is known to have a significant impact on the Kuramoto Model's stability (Acebrón et al., 2005). In this sense, the threshold between drifting and stable state of oscillators occurs from a critical coupling strength that depends on the topology and initial parameters of the network. Determining the critical coupling strength for achieving a synchronized state on complex networks becomes a well-known major challenge (Arenas et al., 2008).

Therefore, COOPT attempts to avoid the impact of such drifting oscillators state by providing a stable and monotonic convergence process, as seen in Section 5.4. Thus, agents run a local search that monotonically decreases the solution cost over time. We run the Friedman test to observe the relevance of the parameter K on the solution costs. The Friedman test was performed on results from random problems involving 10 and 50 variables, and densities between 30% and 90%, with a interval $0.1 \leq K \leq 20$, taking into account 95% of confidence level.

Table 1: Results of the Friedman test for each group of random problems.

(a) 10 variables

Constraints (density)	p-value
30%	0.0501
50%	0.7797
60%	0.1604
70%	0.3998
90%	0.1363

(b) 20 variables

Constraints (density)	p-value
30%	0.8040
50%	0.0981
60%	0.7272
70%	0.2197
90%	0.9010

(c) 30 variables

Constraints (density)	p-value
30%	0.1138
50%	0.9736
60%	0.7014
70%	0.7485
90%	0.0611

(d) 40 variables

Constraints (density)	p-value
30%	0.1289
50%	0.7066
60%	0.7891
70%	0.6273
90%	0.3436

(e) 50 variables

Constraints (density)	p-value
30%	0.4973
50%	0.4563
60%	0.0668
70%	0.9061
90%	0.4447

Table 1 shows the results of the Friedman test for each group of random problems. From the results, we observed that there was no statistically significant difference between each tested value of K regarding the solution cost, producing p-values between 0.0501 and 0.9736. Therefore, we assumed a coupling strength $K = 1$ in our evaluation in order to consider only natural weights from cost functions.

6.5 Results

Tables 2, 3, 4, and 5 show the results of random problems involving 200 variables in regular, random, small-world, and scale-free networks, respectively. Table 6 shows the results of meeting scheduling problems involving 5, 10, and 20 agents per meeting. Results of random problems involving 50 and 100 variables were omitted for readability owing to their similar performance, so we included all omitted results in appendix. We display the Nemenyi ranks for each metric and highlight results by "•" or "◦" when observed a statistically significant improvement or degradation, respectively, taking into account a significance level threshold set at 95%. In addition, the mean values for each metric are displayed in parenthesis.

6.5.1 Random Problems

Regarding the solution cost, COOPT achieved better solutions than the other algorithms in all cases. COOPT exhibited an efficient local computation achieving the highest quality solutions particularly in dense and hard networks. In COOPT, each agent propagates an error degree after taking an action and performing local searches that aims to minimize the solution cost among groups of agents. On the other hand, DSA and DSA-SDP consider only local gains before taking an action; that is, they employ a local optima stochastic strategy. Nevertheless, DSA-SDP calculates the probability of replacing the assignment if there is an

Table 2: Random problems involving 200 variables in regular networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (231967.90) ●	6.00 (1179407.40) ○	3.00 (1207602.00)	3.00 (33163030.33)
	DSA	2.80 (234638.02)	2.00 (131935.80) ●	1.50 (1180000.00) ●	1.00 (26362285.60) ●
	DSA-SDP	2.20 (234225.17) ●	3.00 (362460.60)	1.50 (1180000.00) ●	2.00 (32643759.60) ●
	GDBA	4.00 (235475.32)	5.00 (1148340.60) ○	5.00 (2387814.40) ○	4.00 (61088406.32)
	MGM	5.60 (236697.04) ○	1.00 (71803.00) ●	4.00 (2383600.00)	5.00 (76462659.52) ○
	MGM-2	5.40 (236783.78) ○	4.00 (870024.62)	6.00 (3510373.06) ○	6.00 (105848847.98) ○
50%	COOPT	1.00 (411960.02) ●	6.00 (1988711.10) ○	3.00 (1999602.00)	3.00 (55623903.40)
	DSA	2.70 (415227.36)	2.00 (224171.64) ●	1.50 (1980000.00) ●	1.00 (44249766.66) ●
	DSA-SDP	2.30 (414919.53) ●	3.00 (505721.70)	1.50 (1980000.00) ●	2.00 (54256862.64) ●
	GDBA	4.70 (419594.96) ○	5.00 (1949418.90) ○	4.00 (3979800.00)	4.00 (101917612.56)
	MGM	5.70 (419924.64) ○	1.00 (111632.40) ●	5.00 (3999600.00) ○	5.00 (128286797.76) ○
	MGM-2	4.60 (419511.02) ○	4.00 (1613068.38)	6.00 (5888476.02) ○	6.00 (176905834.06) ○
70%	COOPT	1.00 (595746.68) ●	6.00 (2795442.30) ○	3.00 (2791602.00)	3.00 (78098782.33)
	DSA	2.80 (599854.76)	2.00 (321473.64) ●	1.50 (2780000.00) ●	1.00 (62148178.80) ●
	DSA-SDP	2.20 (599276.90) ●	3.00 (529784.60)	1.50 (2780000.00) ●	2.00 (75856129.84) ●
	GDBA	5.00 (609943.56) ○	5.00 (2748071.70) ○	4.00 (5571782.70)	4.00 (142727014.40)
	MGM	5.70 (610485.70) ○	1.00 (151676.80) ●	5.00 (5615600.00) ○	5.00 (180074789.12) ○
	MGM-2	4.30 (609474.48)	4.00 (2353642.52)	6.00 (8266488.14) ○	6.00 (247900742.14) ○
90%	COOPT	1.00 (781342.80) ●	6.00 (3599600.00) ○	3.00 (3583602.00)	3.00 (100574895.93)
	DSA	2.80 (785428.38)	2.00 (355937.92) ●	1.50 (3580000.00) ●	1.00 (80040179.36) ●
	DSA-SDP	2.20 (784514.10) ●	3.00 (501420.77)	1.50 (3580000.00) ●	2.00 (97472863.44) ●
	GDBA	4.60 (803309.90) ○	5.00 (3544200.00) ○	4.00 (7163781.00)	4.00 (183534999.76)
	MGM	5.80 (804524.48) ○	1.00 (190957.20) ●	5.00 (7231600.00) ○	5.00 (231852972.00) ○
	MGM-2	4.60 (803260.82) ○	4.00 (3149676.84)	6.00 (10644104.64) ○	6.00 (318833019.88) ○

Table 3: Random problems involving 200 variables in random networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (235412.80) ●	6.00 (1191576.90) ○	3.00 (1221462.00)	3.00 (33549175.20)
	DSA	2.40 (238118.82)	2.00 (171816.58) ●	1.50 (1194000.00) ●	1.00 (26666884.72) ●
	DSA-SDP	3.00 (238589.40)	3.00 (256904.70)	1.50 (1194000.00) ●	2.00 (33023970.28) ●
	GDBA	3.60 (238873.02)	5.00 (1157022.90) ○	5.00 (2415696.00) ○	4.00 (61802853.00)
	MGM	5.60 (240493.74) ○	1.00 (93041.20) ●	4.00 (2411880.00)	5.00 (77366215.28) ○
	MGM-2	5.40 (240296.16) ○	4.00 (971128.92)	6.00 (3551974.94) ○	6.00 (107023478.34) ○
50%	COOPT	1.00 (414738.84) ●	6.00 (1993199.60) ○	3.00 (2009502.00)	3.00 (55907192.70)
	DSA	2.80 (418195.46)	2.00 (251950.46) ●	1.50 (1990000.00) ●	1.00 (44469367.58) ●
	DSA-SDP	2.20 (417614.83) ●	3.00 (378683.20)	1.50 (1990000.00) ●	2.00 (54530854.18) ●
	GDBA	4.70 (422110.90) ○	5.00 (1950230.70) ○	4.00 (3999753.20)	4.00 (102428830.98)
	MGM	5.60 (422821.18) ○	1.00 (132121.40) ●	5.00 (4019800.00) ○	5.00 (128934543.40) ○
	MGM-2	4.70 (421999.52) ○	4.00 (1713192.30)	6.00 (5918201.90) ○	6.00 (177785765.90) ○
70%	COOPT	1.00 (596941.64) ●	6.00 (2799291.40) ○	3.00 (2797542.00)	3.00 (78256749.73)
	DSA	2.50 (600728.48)	2.00 (356877.30) ●	1.50 (2786000.00) ●	1.00 (62269396.32) ●
	DSA-SDP	2.50 (600886.63)	3.00 (410977.83)	1.50 (2786000.00) ●	2.00 (76024866.68) ●
	GDBA	4.80 (610978.28) ○	5.00 (2748339.00) ○	4.00 (5583751.80)	4.00 (143035229.72)
	MGM	5.70 (611312.50) ○	1.00 (172352.40) ●	5.00 (5627720.00) ○	5.00 (180461420.24) ○
	MGM-2	4.50 (610438.90) ○	4.00 (2315252.58)	6.00 (8284296.32) ○	6.00 (248378823.18) ○
90%	COOPT	1.00 (781606.48) ●	6.00 (3598571.10) ○	3.00 (3585582.00)	3.00 (100602188.67)
	DSA	2.60 (785787.56)	2.00 (371012.08) ●	1.50 (3582000.00) ●	1.00 (80113636.02) ●
	DSA-SDP	2.40 (785602.33)	3.00 (452966.67)	1.50 (3582000.00) ●	2.00 (97523467.32) ●
	GDBA	4.90 (803961.66) ○	5.00 (3541180.50) ○	4.00 (7167800.00)	4.00 (183639181.40)
	MGM	5.40 (804167.58) ○	1.00 (199838.60) ●	5.00 (7235640.00) ○	5.00 (231982672.24) ○
	MGM-2	4.70 (803836.58) ○	4.00 (3076880.94)	6.00 (10649774.70) ○	6.00 (318681381.28) ○

Table 4: Random problems involving 200 variables in small-world networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (227804.00) •	6.00 (1160210.70) ○	3.00 (1187802.00)	3.00 (32599037.73)
	DSA	2.90 (230606.60)	2.00 (143568.46) •	1.50 (1160000.00) •	1.00 (25911499.90) •
	DSA-SDP	2.50 (230343.93)	3.00 (284770.40)	1.50 (1160000.00) •	2.00 (32107584.12) •
	GDBA	3.70 (230965.02)	5.00 (1127065.50) ○	5.00 (2348008.20) ○	4.00 (60066947.64)
	MGM	5.50 (232437.58) ○	1.00 (83212.80) •	4.00 (2343200.00)	5.00 (75165808.88) ○
	MGM-2	5.40 (232302.18) ○	4.00 (845465.26)	6.00 (3451035.58) ○	6.00 (104148238.34) ○
50%	COOPT	1.00 (407831.32) •	6.00 (1964885.10) ○	3.00 (1979802.00)	3.00 (55060422.97)
	DSA	2.60 (410939.64)	2.00 (237964.74) •	1.50 (1960000.00) •	1.00 (43795866.74) •
	DSA-SDP	2.40 (410527.33)	3.00 (398974.57)	1.50 (1960000.00) •	2.00 (53706114.58) •
	GDBA	5.00 (414976.88) ○	5.00 (1922471.10) ○	4.00 (3940033.30)	4.00 (100896257.38)
	MGM	5.10 (415364.48) ○	1.00 (134583.40) •	5.00 (3959200.00) ○	5.00 (126991597.48) ○
	MGM-2	4.90 (415009.02) ○	4.00 (1507650.08)	6.00 (5829093.84) ○	6.00 (175187527.42) ○
70%	COOPT	1.00 (591250.50) •	6.00 (2773032.30) ○	3.00 (2771802.00)	3.00 (77530062.40)
	DSA	2.60 (595495.46)	2.00 (339088.52) •	1.50 (2760000.00) •	1.00 (61691125.68) •
	DSA-SDP	2.40 (595191.03)	3.00 (511045.67)	1.50 (2760000.00) •	2.00 (75326266.92) •
	GDBA	5.00 (604899.82) ○	5.00 (2722727.70) ○	4.00 (5532009.70)	4.00 (141708861.24)
	MGM	5.30 (605480.18) ○	1.00 (167345.60) •	5.00 (5575200.00) ○	5.00 (178779010.96) ○
	MGM-2	4.70 (604940.22) ○	4.00 (2285119.00)	6.00 (8206987.64) ○	6.00 (246076714.32) ○
90%	COOPT	1.00 (776224.84) •	6.00 (3577563.30) ○	3.00 (3563802.00)	3.00 (99995994.90)
	DSA	2.70 (781444.96)	2.00 (373941.34) •	1.50 (3560000.00) •	1.00 (79596807.92) •
	DSA-SDP	2.30 (780793.15) •	3.00 (474610.10)	1.50 (3560000.00) •	2.00 (96934165.02) •
	GDBA	4.50 (798125.30)	5.00 (3519687.60) ○	4.00 (7124004.60)	4.00 (182516585.02)
	MGM	5.90 (799881.96) ○	1.00 (199462.40) •	5.00 (7191200.00) ○	5.00 (230559610.84) ○
	MGM-2	4.60 (798328.98) ○	4.00 (3079767.92)	6.00 (10584596.70) ○	6.00 (316931798.52) ○

Table 5: Random problems involving 200 variables in scale-free networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (212929.02) •	5.00 (835950.80) ○	3.00 (1101672.00)	3.00 (30109725.37)
	DSA	3.30 (216276.72)	2.00 (267634.10) •	1.50 (1073000.00) •	1.00 (23893580.54) •
	DSA-SDP	3.00 (216213.50)	3.00 (293636.70)	1.50 (1073000.00) •	2.00 (29707349.46) •
	GDBA	3.00 (216085.40)	4.00 (789693.30)	5.00 (2176595.20) ○	4.00 (55647326.10)
	MGM	5.50 (217996.86) ○	1.00 (219084.60) •	4.00 (2167460.00)	5.00 (69505166.68) ○
	MGM-2	5.20 (217240.08) ○	6.00 (2167972.44) ○	6.00 (3192989.32) ○	6.00 (96793307.62) ○
50%	COOPT	1.00 (355924.16) •	5.00 (1382172.30) ○	3.00 (1737252.00)	3.00 (48126264.50)
	DSA	2.90 (359372.18)	2.50 (302561.98)	1.50 (1715000.00) •	1.00 (38235964.72) •
	DSA-SDP	2.40 (359382.37)	2.50 (301926.87)	1.50 (1715000.00) •	2.00 (47044460.06) •
	GDBA	3.80 (360533.54)	4.00 (1331282.70)	4.00 (3454995.00)	4.00 (88466876.86)
	MGM	5.90 (362435.02) ○	1.00 (215669.40) •	5.00 (3464300.00) ○	5.00 (111101155.56) ○
	MGM-2	5.00 (361729.88) ○	6.00 (2496582.82) ○	6.00 (5100692.10) ○	6.00 (153918964.00) ○
70%	COOPT	1.00 (482440.28) •	5.00 (1864904.10) ○	3.00 (2293632.00)	3.00 (63925062.30)
	DSA	2.60 (486501.90)	2.20 (300578.23) •	1.50 (2277000.00) •	1.00 (50799270.24) •
	DSA-SDP	2.40 (486049.10)	2.80 (322163.66)	1.50 (2277000.00) •	2.00 (62232957.86) •
	GDBA	4.90 (490116.74) ○	4.00 (1812115.80)	4.00 (4573774.30)	4.00 (117159910.42)
	MGM	5.80 (491232.88) ○	1.00 (204502.40) •	5.00 (4599540.00) ○	5.00 (147515159.48) ○
	MGM-2	4.30 (489416.78)	6.00 (2722030.52) ○	6.00 (6771309.38) ○	6.00 (203995680.12) ○
90%	COOPT	1.00 (591850.02) •	5.00 (2303279.10) ○	3.00 (2770812.00)	3.00 (77466128.70)
	DSA	2.60 (596518.92)	2.00 (300176.00) •	1.50 (2759000.00) •	1.00 (61607930.96) •
	DSA-SDP	2.40 (596155.00)	3.00 (338654.50)	1.50 (2759000.00) •	2.00 (75252357.54) •
	GDBA	4.80 (604132.30) ○	4.00 (2249250.30)	4.00 (5533489.20)	4.00 (141757005.38)
	MGM	5.40 (604680.72) ○	1.00 (206411.40) •	5.00 (5573180.00) ○	5.00 (178713348.92) ○
	MGM-2	4.80 (604091.44) ○	6.00 (2889677.40) ○	6.00 (8204144.74) ○	6.00 (246509509.84) ○

improving alternative (Zivan et al., 2014), whereas DSA assumes a constant probability of replacing the assignment, since a local improvement is possible (Zhang et al., 2005).

MGM and MGM-2 also employ a local optima-based strategy, but they introduce a monotonic optimization process (Maheswaran et al., 2004a; Pearce et al., 2008). But in MGM-2, agents form groups of one or more until no group of two or fewer agents can improve the solution (Pearce et al., 2008), generally resulting in better solutions in comparison to MGM. GDBA found considerably worse solutions especially in dense networks because it improves solutions more slowly than the other algorithms. Such relative slowness behavior has already been discussed by authors (Okamoto et al., 2016).

Results demonstrated that COOPT, GDBA, and MGM-2 require more non-concurrent constraint checks than the other algorithms in most cases. The reason is that COOPT performs two sequential steps during the search; that is, when agents choose new values and when agents report aggregated costs. Nevertheless, these sequential steps lead the search faster toward high-quality solutions and guarantee a monotonic convergence process, respectively. COOPT also showed slightly better NCCC in scale-free networks because of the existence of hubs that help to reduce the depth of the spanning-tree.

No statistical difference of NCCC between MGM and DSA was observed. In fact, procedures of MGM and DSA are simple and only consider unilateral actions by agents in a given context. MGM-2 require more message cycles and constraint checks per iteration than MGM (Pearce et al., 2008) and thus affecting the NCCC. Because the DSA's communication model does not include a synchronization mechanism (Zhang et al., 2005), the NCCC grew more slowly than the other algorithms, except the MGM that does not always check constraints in a communication round (Maheswaran et al., 2004a).

With respect to the number of exchanged messages, DSA and DSA-SDP require fewer messages than the other algorithms because agents perform only one message cycle per iteration for updating their assignments (Zhang et al., 2005) and providing an anytime search (Zivan et al., 2014). COOPT needs additional messages due to the synchronization stage of the algorithm. Although DSA and DSA-SDP send fewer messages than COOPT, there was no statistically significant difference between them. On the other hand, MGM and GDBA perform two message cycles per iteration for updating assignments and reporting local gains (Maheswaran et al., 2004a; Okamoto et al., 2016). MGM-2 requires five message cycles per iteration (Pearce et al., 2008), resulting in the highest number of exchanged messages in random problems.

Except the MGM-2, all algorithms exchange very little information between agents in each iteration, such as assignments, local gains, accumulated costs, and the index of the best round so far. Nevertheless, agents in MGM-2 send messages containing the suggested values for each agent and the local gain of each value pair, producing the largest messages from experiments. As a consequence, the throughput becomes equivalent to the number of exchanged messages in MGM-2.

6.5.2 Meeting Scheduling Problems

Regarding the meeting scheduling problems, COOPT also achieved better solutions than the other algorithms in all cases. Although the difference between the results was smaller, COOPT achieved the highest quality solutions in large but quite sparse problems. Such

Table 6: Meeting scheduling problems involving 400 variables (CD: 1.377)

Agents per meeting	Algorithm	Solution cost	NCCC	Messages	Throughput
5	COOPT	1.37 (26297.12) ●	4.00 (242992.85)	3.00 (208162.14)	3.00 (5300663.04)
	DSA	3.93 (28707.89) ○	1.10 (130628.80) ●	1.50 (161600.00) ●	1.00 (3599041.96) ●
	DSA-SDP	3.83 (28817.61) ○	2.97 (158790.45)	1.50 (161600.00) ●	2.00 (4499621.29) ●
	GDBA	4.00 (29088.55) ○	5.00 (1167515.93) ○	5.00 (366686.40) ○	4.00 (9126851.18)
	MGM	4.03 (29141.48) ○	1.93 (144262.20) ●	4.00 (323200.00)	5.00 (10369410.24) ○
	MGM-2	3.83 (28549.98) ○	6.00 (1546115.09) ○	6.00 (476255.63) ○	6.00 (17367339.81) ○
10	COOPT	2.00 (29707.66) ●	4.00 (486070.87)	3.00 (413921.27)	3.00 (10836011.02)
	DSA	4.03 (31893.31) ○	1.00 (208543.49) ●	1.50 (363600.00) ●	1.00 (8085171.04) ●
	DSA-SDP	3.23 (31143.91) ●	3.00 (309099.70)	1.50 (363600.00) ●	2.00 (10054462.30) ●
	GDBA	3.37 (31274.02) ●	5.87 (2590044.40) ○	5.00 (770537.25) ○	4.00 (19448946.46)
	MGM	3.80 (31897.45) ○	2.00 (231618.80) ●	4.00 (727200.00)	5.00 (23321700.96) ○
	MGM-2	4.57 (32440.18) ○	5.13 (2364909.11) ○	6.00 (1071060.65) ○	6.00 (37381380.00) ○
20	COOPT	2.10 (29526.99) ●	4.00 (968812.26)	3.00 (820100.45)	3.00 (21780861.80)
	DSA	3.70 (31794.64) ○	1.00 (356964.14) ●	1.50 (767600.00) ●	1.00 (17037329.18) ●
	DSA-SDP	3.23 (31064.55) ●	3.00 (634049.03)	1.50 (767600.00) ●	2.00 (21082288.82) ●
	GDBA	2.67 (30348.55) ●	6.00 (5811894.04) ○	5.00 (1572754.20) ○	4.00 (39951080.08)
	MGM	4.40 (32380.60) ○	2.00 (396283.00) ●	4.00 (1535200.00)	5.00 (49194539.96) ○
	MGM-2	4.90 (34005.00) ○	5.00 (3638752.05) ○	6.00 (2261029.87) ○	6.00 (77011099.61) ○

behavior was mainly observed from experiments involving 5 agents per meeting, where only COOPT presented results statistically significant in terms of solution cost. Unlike the results of random problems, GDBA showed better solutions than DSA in meeting scheduling problems. In fact, sparse problems are favorable to GDBA because it is able to achieve high-quality solutions in few iterations, even improving solutions more slowly than the other algorithms.

As also observed in random problems, DSA and MGM required less NCCC than the other algorithms in meeting scheduling problems. GDBA showed a slightly worse results because in our meeting scheduling problems agents can hold more than one variable, whereas in our random problems agents hold a single variable. Besides, the growth of NCCC in GBDA is also related to the domain size, which was twice larger in meeting scheduling problems than in random problems.

In terms of number of exchanged messages, COOPT also exhibited good results in meeting scheduling problems. Nevertheless, we could observe a small difference between GDBA and MGM. This difference was motivated by the number of variables held by a single agent, as also observed in NCCC. Therefore, GDBA seems to be more sensitive to the number of variables per agent in terms of NCCC and number of messages. Finally, throughput in meeting scheduling problems are very close to random problems, in which MGM-2 produced the highest throughput from experiments.

6.6 Convergence Analysis

One of most notable features of the Kuramoto model is its convergence ability, showing a natural tendency to deal with complex and large networks. Thus, we provide a convergence analysis comparing the solution improvement ability of the local search algorithms over 300 iterations. Figures 7 and 8 show the convergence dynamics of the algorithms on random problems involving sparse (30% of density) and dense (90% of density) constraint networks,

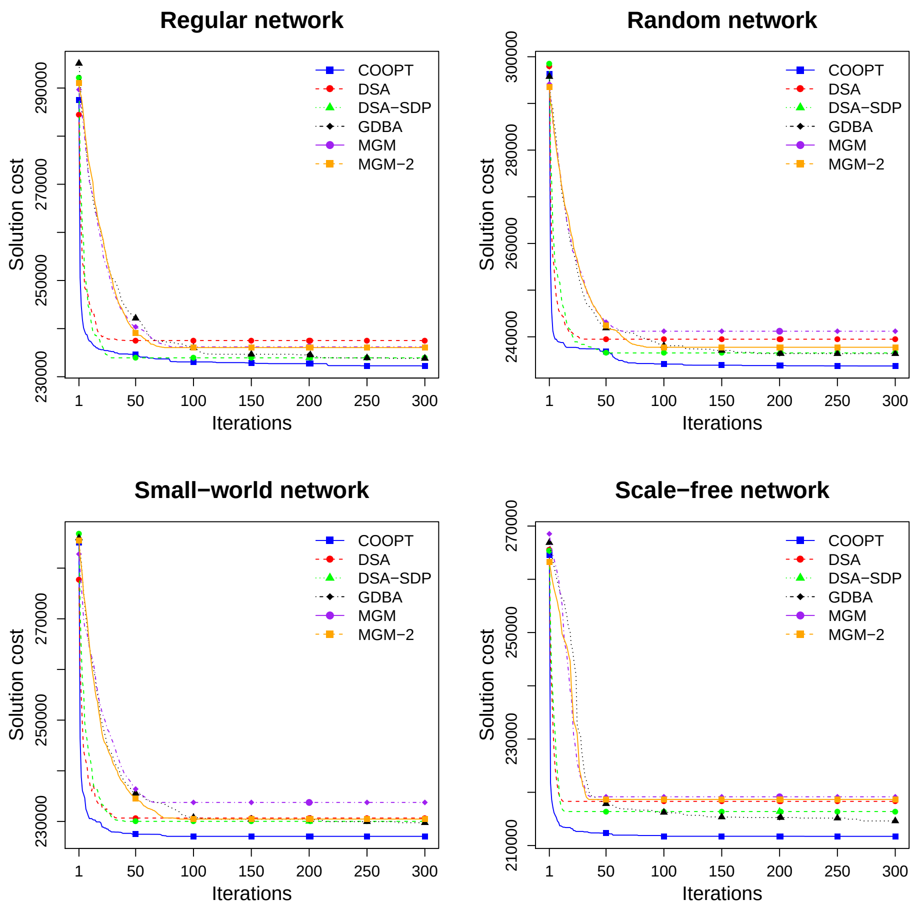


Figure 7: Convergence on sparse random problems.

respectively. Figure 9 shows the convergence dynamics of meeting scheduling problems with 10 and 20 agents per meeting.

6.6.1 Random Problems

From the convergence analysis involving random problems, COOPT exhibited a quick and consistent convergence in relation to the other algorithms, requiring about 100 iterations to achieve high-quality solutions in most cases. After the threshold of 100 iterations, no solution improvements on sparse and dense random problems were observed in COOPT. Moreover, the convergence ability of COOPT seems to be consistent in all the network topologies. Despite the simple strategy of DSA, it found good solutions in few iterations requiring less computational and communication effort. DSA-SDP also exhibited a quick convergence and its anytime approach led the search towards better solutions than DSA.

In contrast, GDBA, MGM, and MGM-2 showed solution improvements significantly slower than COOPT, DSA and DSA-SDP. Such algorithms need much more interations for converging toward high-quality solutions, but we observed that GDBA achieved better

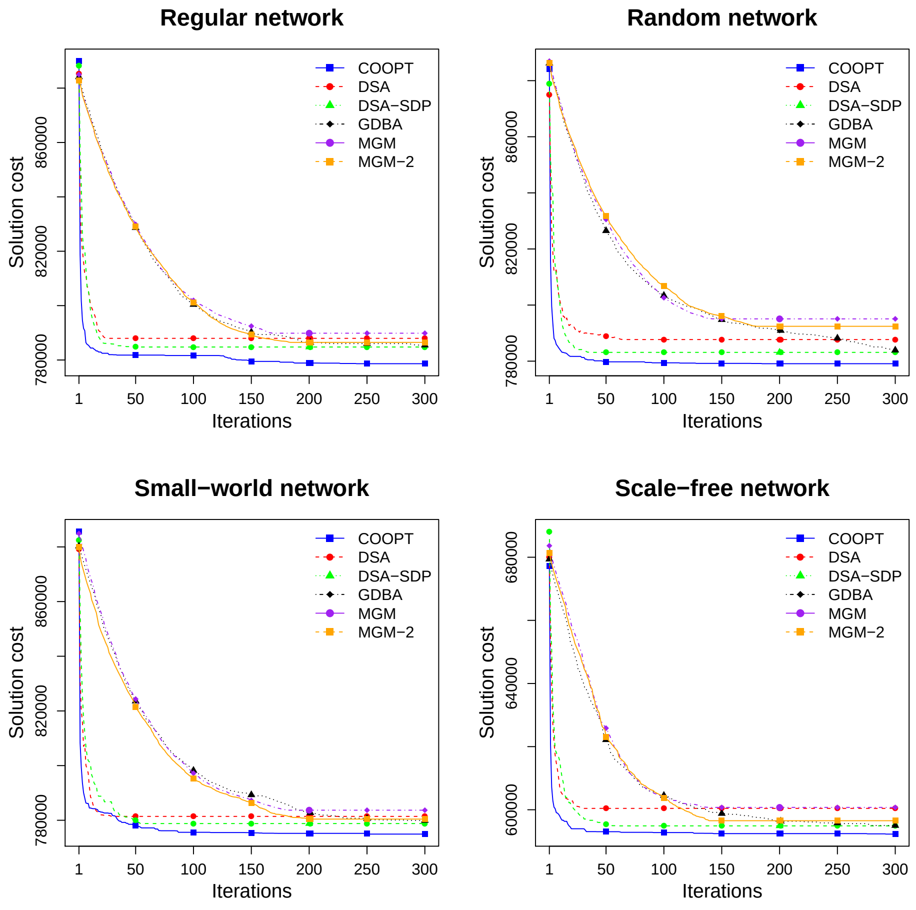


Figure 8: Convergence on dense random problems.

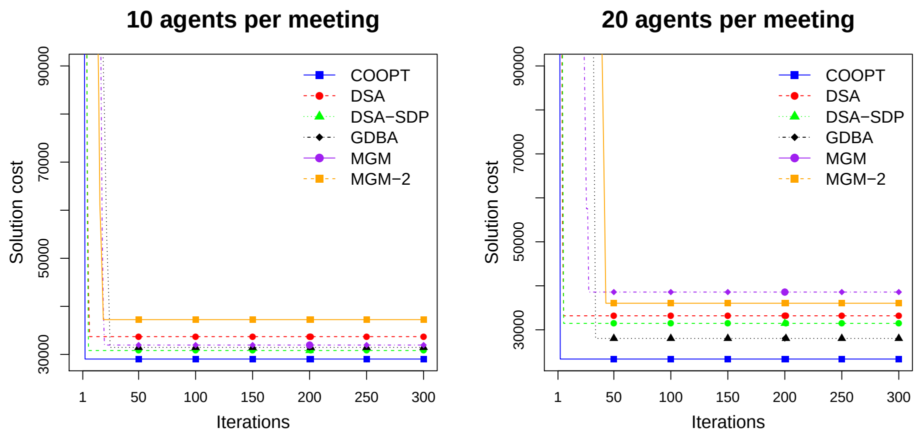


Figure 9: Convergence on meeting scheduling problems.

solutions than DSA-SDP in most of cases. GDBA benefits from extended runtime achieving high-quality solutions on sparse and dense random problems close to 300 iterations. MGM-2 found better solutions than DSA, but requiring much more computational and communication effort, as seen from Tables 2, 3, 4, and 5. Finally, MGM exhibited a slow and poor convergence in all cases, resulting in the worst solutions on random problems.

6.6.2 Meeting Scheduling Problems

Although our meeting scheduling problems have a state space grater than random problems, the constraint density is significantly lower. Such aspect helped all algorithms for achieving high-quality solutions faster. From the results of meeting scheduling problems, we observed that COOPT converged to the highest quality solutions in less than 5 iterations in all cases. DSA-SDP and GDBA exhibited close results, but GDBA required more iterations for achieving high-quality solutions. DSA also showed a very fast convergence, but finding worse solutions than COOPT, DSA-SDP, and GDBA. Finally, MGM-2 and MGM resulted in the worst solutions in meeting scheduling problems.

6.7 Network Topology Analysis

From the results of random problems, we could observe that the degree distribution of the network has a relevant effect on the performance of COOPT. Regular networks have an unique degree for all vertexes (Wasserman & Faust, 1994). Random and small-world networks produce vertexes with degree very similar because the probability of wiring or rewiring is the same for all elements (Erdös & Rényi, 1959; Watts & Strogatz, 1998). However, scale-free networks have a degree distribution following an exponential growth (Barabási & Albert, 1999). Such aspect in scale-free networks contributed to COOPT to achieve high-quality solutions faster by reducing the depth of the spanning-tree, as seen from Figures 7 and 8.

On the other hand, COOPT needed more iterations for achieving high-quality solutions on regular and random networks. Such behavior occurs because all agents have the almost the same amount of the neighbors (Pearce, Maheswaran, & Tambe, 2005); that is, the joint actions are equally spread on the network. Moreover, COOPT was affected by the degree distribution in terms of communication effort, showing a slightly better performance on scale-free networks. Therefore, the degree distribution and depth of the spanning-tree are key factors for COOPT in terms of computational and communication effort.

6.8 Network Density Analysis

Regarding the network density, we could conclude that COOPT is suitable to deal with sparse or over-constrained large-scale problems. Experiments showed a stable convergence ability of COOPT, achieving better results in terms of quality solution in few iterations, even on sparse and dense problems. From the results of random problems, COOPT, DSA, and DSA-SDP require almost the same amount of iterations for achieving high-quality solutions in sparse and dense problems. In contrast, GBDA, MGM, and MGM-2 displayed a quite slow converge towards high-quality solutions in dense problems.

In addition to meeting scheduling problems, which had a constraint density substantially lower than random problems, there was no statistical difference between DSA, DSA-SDP, GDBA, MGM and MGM-2 in relation to the solution quality; that is, only COOPT found the highest quality solutions with statistical significance. In contrast, all the algorithms required much more computational and communication effort in dense problems than in sparse problems. But COOPT can have an advantage by reducing the amount of iterations due to its stable and quick convergence process.

6.9 Discussion

Our experimental evaluation demonstrated that COOPT outperformed the other algorithms in almost all cases. Specifically, COOPT achieved better results than the other algorithms such as DUCT, GDBA, MGM, and MGM-2 in terms of communication load, displaying an efficient communication model. Moreover, no statistical difference between COOPT, DSA and DSA-SDP was observed in terms of number of messages and throughput. We could also observe significant improvements of COOPT regarding the solution quality, particularly on dense and hard problems.

Therefore, COOPT produced better solutions in comparison to the other algorithms, requiring fewer iterations even on dense problems for converging to high-quality solutions. Such behavior demonstrated the convergence ability of COOPT and its potential to handle large-scale, complex applications. Our experimental evaluation also demonstrated that the network topology has a relevant effect on the performance of COOPT, although it has exhibited a stable convergence producing better solutions on several topologies with different degrees distribution.

7. Conclusion

In this paper, we addressed a novel approach for solving DCOP inspired by synchronization of coupled oscillator networks. This approach has introduced the concepts of the Kuramoto model in order to improve the convergence ability and scalability on large-scale DCOPs. We proposed a new incomplete DCOP algorithm, called COOPT, in which agents perform an anytime local search and use an error degree propagation technique to coordinate their local actions.

We provided an empirical evaluation and results showed that COOPT outperformed other state-of-the-art DCOP algorithms, particularly on hard and dense problems. From the results, we observed the advantages of COOPT on the communication load needed by agents. Moreover, COOPT also achieved better solutions than the other algorithms. Such features and its anytime local search is able to improve the scalability on large-scale, complex real applications.

Our experimental evaluation showed that the performance of COOPT can be affected by different network topologies. Nevertheless, COOPT exhibited a stable convergence process producing better solutions on several topologies. Such behavior is an important advantage because it allows using COOPT on complex large-scale applications with different degrees distribution. COOPT showed an efficient communication model, producing better solutions than the other algorithms requiring low communication load.

As future work, we intend to explore multiple lines about COOPT. First, we suggest a deeper analysis of the error degree propagation technique, as well as refinements of the communication model to improve its performance. Still on performance, we suggest studies aimed at improving the stability of the synchronization process and the impact of the K coupling strength on the convergence ability. Studies involving non-binary constraints are also recommended for increasing the applicability of COOPT. Finally, we aim to design a version of COOPT that can adapt to changes, so that it can be applied in complex dynamic environments.

Appendix A. Experiments Involving 50 and 100 Variables

Tables 7 through 14 display the results of random problems involving 50 and 100 variables in regular, random, small-world, and scale-free networks, respectively. Tables show mean values (in parenthesis) and Nemenyi ranks for each metric, highlighting results by "$\bullet$" or "$\circ$" when observed a statistically significant improvement or degradation, respectively, taking into account a significance level threshold set at 95%.

Table 7: Random problems involving 50 variables in regular networks (CD: 1.645)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (10244.96) ●	5.00 (73735.20)	3.00 (79002.00) ●	3.00 (2081691.36)
	DSA	3.10 (10745.08)	2.00 (19260.92) ●	1.50 (70000.00) ●	1.00 (1559831.28) ●
	DSA-SDP	2.70 (10703.78)	3.00 (37748.62)	1.50 (70000.00) ●	2.00 (1970786.44) ●
	DUCT	7.00 (13633.76) ○	7.00 (150090275.84) ○	7.00 (15568691.96) ○	7.00 (2629970782.02) ○
	GDBA	3.20 (10733.48)	4.00 (66250.80)	5.00 (149037.00)	4.00 (3768203.04)
	MGM	5.50 (10955.02) ○	1.00 (16300.20) ●	4.00 (141400.00)	5.00 (4534317.76)
	MGM-2	5.50 (10967.59) ○	6.00 (147017.64) ○	6.00 (209159.31) ○	6.00 (6463747.30) ○
50%	COOPT	1.00 (20210.50) ●	5.00 (125531.20)	3.00 (128502.00) ●	3.00 (3430016.64)
	DSA	3.10 (20770.76)	2.00 (35389.44) ●	1.50 (120000.00) ●	1.00 (2675332.32) ●
	DSA-SDP	2.60 (20781.67) ●	3.00 (72053.76)	1.50 (120000.00) ●	2.00 (3374392.84) ●
	DUCT	7.00 (24958.66) ○	7.00 (277280471.80) ○	7.00 (15801143.80) ○	7.00 (2848689374.80) ○
	GDBA	3.30 (20780.54)	4.00 (115948.80)	5.00 (248554.80)	4.00 (6324438.24)
	MGM	5.70 (21232.40) ○	1.00 (26740.80) ●	4.00 (242400.00)	5.00 (7773179.52)
	MGM-2	5.30 (21150.77)	6.00 (250039.44) ○	6.00 (357535.87) ○	6.00 (10893161.71) ○
70%	COOPT	1.00 (30707.16) ●	5.00 (178960.80)	3.00 (178002.00) ●	3.00 (4781903.84)
	DSA	3.10 (31435.04)	2.00 (45953.72) ●	1.50 (170000.00) ●	1.00 (3790388.68) ●
	DSA-SDP	2.70 (31411.10)	3.00 (91424.64)	1.50 (170000.00) ●	2.00 (4777054.28) ●
	DUCT	7.00 (36348.08) ○	7.00 (417001503.50) ○	7.00 (15923599.64) ○	7.00 (2927864043.82) ○
	GDBA	3.20 (31393.52)	4.00 (167290.20)	5.00 (348042.80)	4.00 (8879239.68)
	MGM	5.60 (31885.10) ○	1.00 (36716.60) ●	4.00 (343400.00)	5.00 (11012246.40)
	MGM-2	5.40 (31831.26) ○	6.00 (356954.44) ○	6.00 (506047.62) ○	6.00 (15317580.55) ○
90%	COOPT	1.00 (41433.16) ●	5.00 (230609.20)	3.00 (227502.00) ●	3.00 (6181110.92)
	DSA	3.00 (42107.63)	2.00 (53966.44) ●	1.50 (220000.00) ●	1.00 (4906130.24) ●
	DSA-SDP	2.80 (42166.81)	3.00 (94486.92)	1.50 (220000.00) ●	2.00 (6130776.64) ●
	DUCT	7.00 (47975.08) ○	7.00 (545471867.20) ○	7.00 (15984558.00) ○	7.00 (2958792917.14) ○
	GDBA	3.20 (42142.10)	4.00 (216928.80)	5.00 (447554.80)	4.00 (11435039.52)
	MGM	6.00 (42846.60) ○	1.00 (46411.20) ●	4.00 (444400.00)	5.00 (14250917.12)
	MGM-2	5.00 (42621.80)	6.00 (473272.36) ○	6.00 (654610.93) ○	6.00 (19745400.82) ○

Table 8: Random problems involving 50 variables in random networks (CD: 1.645)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (10847.44) ●	5.00 (75047.20)	3.00 (82368.00) ●	3.00 (2173549.60)
	DSA	3.40 (11376.84)	2.00 (36504.80) ●	1.50 (73400.00) ●	1.00 (1634939.22) ●
	DSA-SDP	2.70 (11353.45)	3.00 (52750.00)	1.50 (73400.00) ●	2.00 (2065900.09) ●
	DUCT	7.00 (14015.28) ○	7.00 (114433184.82) ○	7.00 (14975384.40) ○	7.00 (2351808921.42) ○
	GDBA	3.20 (11371.22)	4.00 (66250.80)	5.00 (155829.40)	4.00 (3942715.20)
	MGM	5.40 (11602.97) ○	1.00 (26648.90) ●	4.00 (148268.00)	5.00 (4754491.20)
	MGM-2	5.30 (11619.65)	6.00 (210018.31) ○	6.00 (219275.06) ○	6.00 (6771894.96) ○
50%	COOPT	1.00 (20831.14) ●	5.00 (128468.20)	3.00 (130878.00) ●	3.00 (3495662.12)
	DSA	3.40 (21407.56)	2.00 (46478.51) ●	1.50 (122400.00) ●	1.00 (2727368.85) ●
	DSA-SDP	3.00 (21325.17)	3.00 (78581.41)	1.50 (122400.00) ●	2.00 (3441073.85) ●
	DUCT	7.00 (25283.12) ○	7.00 (249083796.40) ○	7.00 (15713523.44) ○	7.00 (2772892700.64) ○
	GDBA	2.60 (21361.12) ●	4.00 (117384.30)	5.00 (253331.90)	4.00 (6446239.32)
	MGM	5.60 (21733.95) ○	1.00 (36486.00) ●	4.00 (247248.00)	5.00 (7928498.16)
	MGM-2	5.40 (21761.16) ○	6.00 (314091.41) ○	6.00 (364665.38) ○	6.00 (11106013.70) ○
70%	COOPT	1.10 (31156.54) ●	5.00 (178967.30)	3.00 (179388.00) ●	2.80 (4818807.22)
	DSA	3.40 (31722.29)	2.00 (60348.83) ●	1.50 (171400.00) ●	1.00 (3822048.74) ●
	DSA-SDP	2.80 (31675.39)	3.00 (91744.83)	1.50 (171400.00) ●	2.20 (4816475.70) ●
	DUCT	7.00 (36443.60) ○	7.00 (373708990.96) ○	7.00 (15881446.68) ○	7.00 (2889563758.18) ○
	GDBA	2.90 (31668.84)	4.00 (166003.20)	5.00 (350847.10)	4.00 (8951282.98)
	MGM	5.40 (32182.14) ○	1.00 (44620.70) ●	4.00 (346228.00)	5.00 (11102750.84)
	MGM-2	5.40 (32162.73) ○	6.00 (412446.06) ○	6.00 (510223.35) ○	6.00 (15455846.39) ○
90%	COOPT	1.00 (41679.82) ●	5.00 (229509.00)	3.00 (227898.00) ●	3.00 (6192176.63)
	DSA	3.10 (42293.91)	2.00 (56330.60) ●	1.50 (220400.00) ●	1.00 (4914473.41) ●
	DSA-SDP	2.90 (42323.62)	3.00 (96003.72)	1.50 (220400.00) ●	2.00 (6141604.92) ●
	DUCT	7.00 (48034.54) ○	7.00 (509699568.52) ○	7.00 (15957773.76) ○	7.00 (2943782523.44) ○
	GDBA	3.30 (42389.34)	4.00 (215087.40)	5.00 (448368.40)	4.00 (11455390.84)
	MGM	5.40 (42839.38) ○	1.00 (49778.80) ●	4.00 (445208.00)	5.00 (14276702.40)
	MGM-2	5.30 (42852.26)	6.00 (501457.80) ○	6.00 (655839.77) ○	6.00 (19804620.48) ○

Table 9: Random problems involving 50 variables in small-world networks (CD: 1.645)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
	COOPT	1.00 (10219.60) •	5.00 (71102.60)	3.00 (79002.00) •	3.00 (2081377.74)
	DSA	3.00 (10703.70)	2.00 (29064.41) •	1.50 (70000.00) •	1.00 (1559640.33) •
	DSA-SDP	2.80 (10738.51)	3.00 (48696.67)	1.50 (70000.00) •	2.00 (1970181.05) •
30%	DUCT	7.00 (13351.86) ◦	7.00 (115800127.60) ◦	7.00 (15105413.64) ◦	7.00 (2376204156.64) ◦
	GDBA	3.40 (10725.46)	4.00 (62983.80)	5.00 (149052.70)	4.00 (3768774.14)
	MGM	5.50 (10959.58) ◦	1.00 (21516.60) •	4.00 (141400.00)	5.00 (4534297.40)
	MGM-2	5.30 (10938.47)	6.00 (179796.88) ◦	6.00 (209182.30) ◦	6.00 (6470311.30) ◦
	COOPT	1.10 (20262.08) •	5.00 (123579.80)	3.00 (128502.00) •	3.00 (3430822.30)
	DSA	2.80 (20813.50)	2.00 (42825.55) •	1.50 (120000.00) •	1.00 (2674457.20) •
	DSA-SDP	2.60 (20816.14) •	3.00 (77240.83)	1.50 (120000.00) •	2.00 (3373850.89) •
50%	DUCT	7.00 (24611.60) ◦	7.00 (236850256.06) ◦	7.00 (15654531.64) ◦	7.00 (2746035828.44) ◦
	GDBA	3.70 (20919.58)	4.00 (112869.90)	5.00 (248561.60)	4.00 (6324148.70)
	MGM	5.40 (21169.85) ◦	1.00 (32776.60) •	4.00 (242400.00)	5.00 (7773379.96)
	MGM-2	5.40 (21159.73) ◦	6.00 (298119.93) ◦	6.00 (357543.21) ◦	6.00 (10903487.88) ◦
	COOPT	1.00 (30780.80) •	5.00 (178547.50)	3.00 (178002.00) •	2.90 (4781924.14)
	DSA	2.80 (31445.25)	2.00 (49736.80) •	1.50 (170000.00) •	1.00 (3789473.02) •
	DSA-SDP	3.30 (31465.25)	3.00 (92802.05)	1.50 (170000.00) •	2.10 (4778387.71) •
70%	DUCT	7.00 (36279.52) ◦	7.00 (381179667.42) ◦	7.00 (15894345.40) ◦	7.00 (2903896845.96) ◦
	GDBA	3.20 (31492.08)	4.00 (165755.70)	5.00 (348053.90)	4.00 (8879643.58)
	MGM	5.50 (31899.12) ◦	1.00 (42998.40) •	4.00 (343400.00)	5.00 (11012228.20)
	MGM-2	5.20 (31847.00)	6.00 (401984.50) ◦	6.00 (506079.39) ◦	6.00 (15331861.15) ◦
	COOPT	1.00 (41406.06) •	5.00 (230320.50)	3.00 (227502.00) •	3.00 (6181261.62)
	DSA	3.20 (42191.80)	2.00 (55897.75) •	1.50 (220000.00) •	1.00 (4905995.59) •
	DSA-SDP	2.60 (42159.66) •	3.00 (91734.92)	1.50 (220000.00) •	2.00 (6130160.92) •
90%	DUCT	7.00 (47938.36) ◦	7.00 (515744329.42) ◦	7.00 (15970429.32) ◦	7.00 (2950671838.16) ◦
	GDBA	3.20 (42219.80)	4.00 (215869.50)	5.00 (447560.50)	4.00 (11435352.68)
	MGM	5.70 (42846.80) ◦	1.00 (50046.70) •	4.00 (444400.00)	5.00 (14250994.32)
	MGM-2	5.30 (42710.45)	6.00 (497236.99) ◦	6.00 (654646.04) ◦	6.00 (19768350.50) ◦

Table 10: Random problems involving 50 variables in scale-free networks (CD: 1.645)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
	COOPT	1.00 (9627.20) •	4.70 (57352.41)	3.00 (73458.00) •	3.00 (1927710.64)
	DSA	2.70 (10038.20)	1.60 (47638.10) •	1.50 (64400.00) •	1.00 (1431748.78) •
	DSA-SDP	2.70 (10024.76)	3.20 (53462.39)	1.50 (64400.00) •	2.00 (1811018.62) •
30%	DUCT	6.90 (10771.40) ◦	7.00 (27604112.94) ◦	7.00 (8928024.56) ◦	7.00 (867631973.32) ◦
	GDBA	4.10 (10117.68)	4.10 (56999.40)	5.00 (137991.70)	4.00 (3483516.00)
	MGM	5.00 (10234.10)	1.40 (46262.70) •	4.00 (130088.00)	5.00 (4170520.08)
	MGM-2	5.60 (10328.01) ◦	6.00 (372904.13) ◦	6.00 (193248.04) ◦	6.00 (6068586.63) ◦
	COOPT	1.00 (17381.26) •	5.00 (89320.60)	3.00 (113058.00) •	3.00 (3007419.22)
	DSA	3.80 (18081.32)	2.00 (59268.56) •	1.50 (104400.00) •	1.00 (2323129.85) •
	DSA-SDP	2.60 (17880.95) •	3.00 (67098.61)	1.50 (104400.00) •	2.00 (2933392.03) •
50%	DUCT	7.00 (19489.28) ◦	7.00 (63528101.28) ◦	7.00 (11092286.68) ◦	7.00 (1322404475.84) ◦
	GDBA	3.00 (17973.26)	4.00 (77279.40)	5.00 (217664.60)	4.00 (5529724.34)
	MGM	5.40 (18310.81) ◦	1.00 (51885.50) •	4.00 (210888.00)	5.00 (6761491.88)
	MGM-2	5.20 (18279.39)	6.00 (437583.72) ◦	6.00 (311626.68) ◦	6.00 (9585078.65) ◦
	COOPT	1.00 (24460.74) •	5.00 (123515.10)	3.00 (147708.00) •	3.00 (3952159.48)
	DSA	3.10 (25125.32)	2.00 (59176.95) •	1.50 (139400.00) •	1.00 (3104514.19) •
	DSA-SDP	3.00 (25084.91)	3.00 (69538.93)	1.50 (139400.00) •	2.00 (3916401.42) •
70%	DUCT	7.00 (27608.82) ◦	7.00 (123059581.88) ◦	7.00 (12669431.32) ◦	7.00 (1727555179.02) ◦
	GDBA	3.00 (25079.18)	4.00 (110553.30)	5.00 (287332.80)	4.00 (7319282.68)
	MGM	5.10 (25443.96)	1.00 (50605.20) •	4.00 (281588.00)	5.00 (9029166.80)
	MGM-2	5.80 (25536.71) ◦	6.00 (464475.79) ◦	6.00 (415343.71) ◦	6.00 (12650919.00) ◦
	COOPT	1.00 (30919.58) •	5.00 (149799.30)	3.00 (177408.00) •	3.00 (4764218.08)
	DSA	2.90 (31527.75)	2.10 (57109.97) •	1.50 (169400.00) •	1.00 (3773885.13) •
	DSA-SDP	3.10 (31521.91)	2.90 (65788.30)	1.50 (169400.00) •	2.00 (4758904.86) •
90%	DUCT	7.00 (34594.46) ◦	7.00 (174351723.32) ◦	7.00 (13637144.08) ◦	7.00 (2024122679.08) ◦
	GDBA	3.00 (31551.90)	4.00 (136491.30)	5.00 (347066.00)	4.00 (8853264.16)
	MGM	5.80 (32050.05) ◦	1.00 (50558.20) •	4.00 (342188.00)	5.00 (10972197.76)
	MGM-2	5.20 (31952.52)	6.00 (483447.10) ◦	6.00 (504404.98) ◦	6.00 (15316777.86) ◦

Table 11: Random problems involving 100 variables in regular networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (50893.02) •	5.00 (293793.00) ○	3.00 (306702.00)	3.00 (8217007.60)
	DSA	3.00 (52027.55)	2.00 (48367.07) •	1.50 (290000.00) •	1.00 (6466989.85) •
	DSA-SDP	2.90 (51919.47)	3.00 (122471.83)	1.50 (290000.00) •	2.00 (8153347.13) •
	GDBA	3.10 (51965.38)	4.00 (278487.00)	5.00 (596789.00) ○	4.00 (15209120.56)
	MGM	5.60 (52862.91) ○	1.00 (33294.90) •	4.00 (585800.00)	5.00 (18787250.96) ○
	MGM-2	5.40 (52879.38) ○	6.00 (319794.02) ○	6.00 (863565.33) ○	6.00 (26240033.47) ○
50%	COOPT	1.00 (93876.04) •	5.00 (496286.70) ○	3.00 (504702.00)	3.00 (13767342.87)
	DSA	3.30 (95458.32)	2.00 (68232.50) •	1.50 (490000.00) •	1.00 (10931500.16) •
	DSA-SDP	2.40 (95169.97)	3.00 (194358.50)	1.50 (490000.00) •	2.00 (13619675.76) •
	GDBA	3.40 (95500.06)	4.00 (476853.30)	5.00 (994814.70) ○	4.00 (25431270.08)
	MGM	5.50 (96387.56) ○	1.00 (52185.00) •	4.00 (989800.00)	5.00 (31743936.56) ○
	MGM-2	5.40 (96400.48) ○	6.00 (561438.08) ○	6.00 (1457870.44) ○	6.00 (43967694.05) ○
70%	COOPT	1.00 (138592.68) •	5.00 (705314.20) ○	3.00 (702702.00)	3.00 (19382561.80)
	DSA	2.80 (139978.40)	2.00 (99589.08) •	1.50 (690000.00) •	1.00 (15396769.71) •
	DSA-SDP	2.30 (139694.23) •	3.00 (227237.70)	1.50 (690000.00) •	2.00 (19020464.40) •
	GDBA	4.20 (140532.40)	4.00 (681733.80)	4.00 (1392784.80)	4.00 (35645767.92)
	MGM	5.80 (141443.20) ○	1.00 (71773.80) •	5.00 (1393800.00) ○	5.00 (44698845.84) ○
	MGM-2	4.90 (141159.31) ○	6.00 (837697.95) ○	6.00 (2052375.83) ○	6.00 (61766261.14) ○
90%	COOPT	1.00 (183634.26) •	5.00 (908700.00) ○	3.00 (900702.00)	3.00 (24996618.07)
	DSA	2.80 (185322.63)	2.00 (107387.40) •	1.50 (890000.00) •	1.00 (19856912.82) •
	DSA-SDP	2.20 (185075.53) •	3.00 (212793.07)	1.50 (890000.00) •	2.00 (24419001.84) •
	GDBA	4.40 (186308.20) ○	4.00 (881100.00)	4.00 (1790791.00)	4.00 (45851607.60)
	MGM	5.40 (186845.27) ○	1.00 (91687.80) •	5.00 (1797800.00) ○	5.00 (57655196.80) ○
	MGM-2	5.20 (186788.62) ○	6.00 (1138919.65) ○	6.00 (2646923.83) ○	6.00 (79560609.98) ○

Table 12: Random problems involving 100 variables in random networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (52376.84) •	5.00 (295780.10) ○	3.00 (313632.00)	3.00 (8404608.40)
	DSA	3.20 (53615.14)	2.00 (70745.13) •	1.50 (297000.00) •	1.00 (6621786.24) •
	DSA-SDP	2.70 (53425.80)	3.00 (103831.73)	1.50 (297000.00) •	2.00 (8347699.40) •
	GDBA	3.10 (53539.40)	4.00 (278477.10)	5.00 (610764.60) ○	4.00 (15567788.88)
	MGM	5.50 (54394.01) ○	1.00 (45854.30) •	4.00 (599940.00)	5.00 (19240424.48) ○
	MGM-2	5.50 (54409.26) ○	6.00 (413262.68) ○	6.00 (884430.37) ○	6.00 (26875696.02) ○
50%	COOPT	1.00 (94911.84) •	5.00 (500782.70) ○	3.00 (509652.00)	3.00 (13906394.40)
	DSA	2.90 (96434.28)	2.00 (86021.74) •	1.50 (495000.00) •	1.00 (11039671.52) •
	DSA-SDP	3.00 (96246.10)	3.00 (164997.83)	1.50 (495000.00) •	2.00 (13753921.48) •
	GDBA	3.20 (96398.92)	4.00 (479031.30)	5.00 (1004778.80) ○	4.00 (25686341.00)
	MGM	5.60 (97514.82) ○	1.00 (64848.70) •	4.00 (999900.00)	5.00 (32067332.80) ○
	MGM-2	5.30 (97383.01) ○	6.00 (648265.24) ○	6.00 (1472763.02) ○	6.00 (44422473.45) ○
70%	COOPT	1.00 (139324.18) •	5.00 (703809.20) ○	3.00 (705672.00)	3.00 (19469578.13)
	DSA	2.80 (140857.94)	2.00 (101049.87) •	1.50 (693000.00) •	1.00 (15457616.00) •
	DSA-SDP	2.60 (140712.23)	3.00 (204505.70)	1.50 (693000.00) •	2.00 (19102121.80) •
	GDBA	3.60 (141133.92)	4.00 (677932.20)	4.00 (1398808.90)	4.00 (35798917.08)
	MGM	5.70 (142123.95) ○	1.00 (84142.50) •	5.00 (1399860.00) ○	5.00 (44893717.04) ○
	MGM-2	5.30 (141996.09) ○	6.00 (911382.89) ○	6.00 (2061337.67) ○	6.00 (62080034.90) ○
90%	COOPT	1.00 (183722.64) •	5.00 (907011.10) ○	3.00 (901692.00)	3.00 (25024846.97)
	DSA	2.60 (185376.61)	2.00 (111863.07) •	1.50 (891000.00) •	1.00 (19876373.01) •
	DSA-SDP	2.40 (185308.13)	3.00 (205275.03)	1.50 (891000.00) •	2.00 (24445243.86) •
	GDBA	4.80 (186858.60) ○	4.00 (878120.10)	4.00 (1792814.30)	4.00 (45904046.66)
	MGM	5.30 (187133.38) ○	1.00 (98468.20) •	5.00 (1799820.00) ○	5.00 (57720698.60) ○
	MGM-2	4.90 (186962.69) ○	6.00 (1155421.09) ○	6.00 (2649859.53) ○	6.00 (79575324.51) ○

Table 13: Random problems involving 100 variables in small-world networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (48757.18) ●	5.00 (281783.60) ○	3.00 (296802.00)	3.00 (7948277.92)
	DSA	2.50 (49958.01)	2.00 (57962.20) ●	1.50 (280000.00) ●	1.00 (6241714.99) ●
	DSA-SDP	2.90 (49981.77)	3.00 (105706.50)	1.50 (280000.00) ●	2.00 (7871193.67) ●
	GDBA	3.60 (50089.76)	4.00 (265221.00)	5.00 (576904.80) ○	4.00 (14696574.72)
	MGM	5.60 (50758.98) ○	1.00 (45112.90) ●	4.00 (565600.00)	5.00 (18139393.04) ○
	MGM-2	5.40 (50674.35) ○	6.00 (367754.49) ○	6.00 (833896.18) ○	6.00 (25343223.64) ○
50%	COOPT	1.00 (92003.62) ●	5.00 (487786.50) ○	3.00 (494802.00)	3.00 (13486841.80)
	DSA	3.40 (93481.92)	2.00 (78667.86) ●	1.50 (480000.00) ●	1.00 (10703630.34) ●
	DSA-SDP	2.70 (93212.43)	3.00 (162223.47)	1.50 (480000.00) ●	2.00 (13345867.76) ●
	GDBA	2.90 (93336.86)	4.00 (466735.50)	5.00 (974914.60) ○	4.00 (24920616.72)
	MGM	5.50 (94305.21) ○	1.00 (64557.10) ●	4.00 (969600.00)	5.00 (31096284.72) ○
	MGM-2	5.50 (94293.47) ○	6.00 (617940.69) ○	6.00 (1428201.02) ○	6.00 (43114321.33) ○
70%	COOPT	1.00 (136095.84) ●	5.00 (693208.30) ○	3.00 (692802.00)	3.00 (19103754.27)
	DSA	2.80 (137799.35)	2.00 (93217.60) ●	1.50 (680000.00) ●	1.00 (15169753.68) ●
	DSA-SDP	2.30 (137571.77) ●	3.00 (201389.83)	1.50 (680000.00) ●	2.00 (18748200.28) ●
	GDBA	3.90 (138058.54)	4.00 (668042.10)	4.00 (1372906.30)	4.00 (35135420.52)
	MGM	5.60 (138850.82) ○	1.00 (81992.90) ●	5.00 (1373600.00) ○	5.00 (44052981.20) ○
	MGM-2	5.40 (138803.72) ○	6.00 (874258.53) ○	6.00 (2022637.21) ○	6.00 (60844798.89) ○
90%	COOPT	1.00 (181038.62) ●	5.00 (894675.30) ○	3.00 (890802.00)	3.00 (24716616.40)
	DSA	2.80 (182988.25)	2.00 (110342.68) ●	1.50 (880000.00) ●	1.00 (19634676.79) ●
	DSA-SDP	2.20 (182695.27) ●	3.00 (203026.83)	1.50 (880000.00) ●	2.00 (24146499.48) ●
	GDBA	4.10 (183959.02)	4.00 (865863.90)	4.00 (1770932.30)	4.00 (45343205.80)
	MGM	5.40 (184541.00) ○	1.00 (99321.80) ●	5.00 (1777600.00) ○	5.00 (57007555.52) ○
	MGM-2	5.50 (184574.89) ○	6.00 (1148696.12) ○	6.00 (2617153.40) ○	6.00 (78606826.84) ○

Table 14: Random problems involving 100 variables in scale-free networks (CD: 1.377)

Density	Algorithm	Solution cost	NCCC	Messages	Throughput
30%	COOPT	1.00 (45901.88) ●	5.00 (208663.10) ○	3.00 (276012.00)	3.00 (7368601.92)
	DSA	3.00 (47086.52)	2.00 (113105.31) ●	1.50 (259000.00) ●	1.00 (5761026.92) ●
	DSA-SDP	2.80 (47011.33)	3.00 (127795.63)	1.50 (259000.00) ●	2.00 (7272646.20) ●
	GDBA	3.60 (47283.28)	4.00 (186288.30)	5.00 (535519.20) ○	4.00 (13628175.36)
	MGM	5.50 (47820.56) ○	1.00 (105318.70) ●	4.00 (523180.00)	5.00 (16774233.20) ○
	MGM-2	5.10 (47712.23) ○	6.00 (882314.74) ○	6.00 (772457.32) ○	6.00 (23635110.34) ○
50%	COOPT	1.00 (79780.12) ●	5.00 (344134.60) ○	3.00 (435402.00)	3.00 (11790384.90)
	DSA	3.40 (81476.87)	2.00 (122548.12) ●	1.50 (420000.00) ●	1.00 (9351414.88) ●
	DSA-SDP	2.80 (81198.23)	3.00 (137398.03)	1.50 (420000.00) ●	2.00 (11713895.50) ●
	GDBA	3.20 (81426.42)	4.00 (319275.00)	5.00 (856120.00) ○	4.00 (21863250.38)
	MGM	5.50 (82207.46) ○	1.00 (107646.20) ●	4.00 (848400.00)	5.00 (27204320.68) ○
	MGM-2	5.10 (82090.72) ○	6.00 (996594.08) ○	6.00 (1250297.16) ○	6.00 (37908740.82) ○
70%	COOPT	1.00 (110608.70) ●	5.00 (471477.00) ○	3.00 (574992.00)	3.00 (15752126.33)
	DSA	3.00 (111979.90)	2.00 (121982.30) ●	1.50 (561000.00) ●	1.00 (12498970.76) ●
	DSA-SDP	3.00 (111897.43)	3.00 (144226.63)	1.50 (561000.00) ●	2.00 (15526493.84) ●
	GDBA	3.20 (112024.42)	4.00 (445262.40)	5.00 (1136878.50) ○	4.00 (29075187.52)
	MGM	5.50 (113162.57) ○	1.00 (101671.50) ●	4.00 (1133220.00)	5.00 (36339205.44) ○
	MGM-2	5.30 (113156.10) ○	6.00 (1071909.62) ○	6.00 (1669143.06) ○	6.00 (50445145.44) ○
90%	COOPT	1.00 (137074.76) ●	5.00 (580087.30) ○	3.00 (694782.00)	3.00 (19150377.70)
	DSA	3.00 (138714.03)	2.00 (118601.79) ●	1.50 (682000.00) ●	1.00 (15206902.35) ●
	DSA-SDP	2.20 (138548.13) ●	3.00 (146817.73)	1.50 (682000.00) ●	2.00 (18793675.02) ●
	GDBA	3.80 (139003.50)	4.00 (553221.90)	4.90 (1377772.00) ○	4.00 (35259812.94)
	MGM	5.80 (139966.06) ○	1.00 (101567.50) ●	4.10 (1377640.00)	5.00 (44179324.04) ○
	MGM-2	5.20 (139901.48) ○	6.00 (1118718.22) ○	6.00 (2028691.92) ○	6.00 (61134031.00) ○

References

Acebrón, J. A., Bonilla, L. L., Pérez-Vicente, C. J., Ritort, F., & Spigler, R. (2005). The kuramoto model: A simple paradigm for synchronization phenomena. *Rev. Mod. Phys.*, *77*, 137–185.

Arenas, A., Daz-Guilera, A., Kurths, J., Moreno, Y., & Zhou, C. (2008). Synchronization in complex networks. *Phys. Rep.*, *469*, 93–153.

Arenas, A., Daz-Guilera, A., & Prez-Vicente, C. J. (2006). Synchronization processes in complex networks. *Physica D: Nonlinear Phenomena*, *224*(1), 27–34. Dynamics on Complex Networks and Applications.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Buck, J., & Buck, E. (1976). Synchronous fireflies. *Scientific American*, *234*, 74–85.

Dörfler, F., M., C., & F., B. (2013). Synchronization in complex oscillator networks and smart grids. *Proceedings of the National Academy of Sciences*, *110*(6), 2005–2010.

Erdös, P., & Rényi, A. (1959). On random graphs i. *Publicationes Mathematicae Debrecen*, *6*, 290.

Farinelli, A., Rogers, A., & Jennings, N. R. (2014). Agent-based decentralised coordination for sensor networks using the max-sum algorithm. *Autonomous Agents and Multi-Agent Systems*, *28*(3), 337–380.

Fioretto, F., Pontelli, E., & Yeoh, W. (2018). Distributed constraint optimization problems and applications: A survey. *J. Artif. Intell. Res.*, *61*, 623–698.

Fitzpatrick, S., & Meertens, L. (2003). *Distributed Coordination through Anarchic Optimization*, pp. 257–295. Springer US, Boston, MA.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, *11*(1), 86–92.

Gaston, M. E., & DesJardins, M. (2005). Agent-organized networks for multi-agent production and exchange. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1*, AAAI'05, pp. 77–82. AAAI Press.

Gushchin, A., Mallada, E., & Tang, A. (2015). Synchronization of phase-coupled oscillators with plastic coupling strength.. In *ITA*, pp. 291–300. IEEE.

Hong, H., & Strogatz, S. H. (2011). Kuramoto model of coupled oscillators with positive and negative coupling parameters: An example of conformist and contrarian oscillators. *Physical Review Letters*, *106*(5), 102–104.

Kiekintveld, C., Yin, Z., Kumar, A., & Tambe, M. (2010). Asynchronous algorithms for approximate distributed constraint optimization with quality bounds.. In van der Hoek, W., Kaminka, G. A., Lesprance, Y., Luck, M., & Sen, S. (Eds.), *AAMAS*, pp. 133–140. IFAAMAS.

Kuramoto, Y. (1975). Self-entrainment of a population of coupled nonlinear oscillators. In Araki, H. (Ed.), *International Symposium on Mathematical Problems in Theoret-*

ical Physics, Lecture Notes in Physics, Vol. 39,, pp. 420–422, New York, NY, USA. Springer.

Kuramoto, Y. (1984). *Chemical oscillations, waves, and turbulence.* Springer-Verlag, New York, NY, USA.

Le, T., Fioretto, F., Yeoh, W., Son, T. C., & Pontelli, E. (2016). Er-dcops: A framework for distributed constraint optimization with uncertainty in constraint utilities. In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '16, pp. 606–614, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Le, T., Son, T. C., Pontelli, E., & Yeoh, W. (2015). Solving distributed constraint optimization problems using logic programming. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 1174–1181. AAAI Press.

Léauté, T., Ottens, B., & Szymanek, R. (2009). Frodo 2.0: An open-source framework for distributed constraint optimization. In *Proceedings of the IJCAI'09 Distributed Constraint Reasoning Workshop (DCR'09)*, pp. 160–164, Pasadena, California, USA. AAAI Press.

Leite, A. R., Enembreck, F., & Barthès, J.-P. A. (2014). Distributed constraint optimization problems: Review and perspectives. *Expert Systems with Applications*, *41*(11), 5139–5157.

Leite, A. R., Giacomet, B., & Enembreck, F. (2009). Railroad driving model based on distributed constraint optimization.. In *IAT*, pp. 474–481. IEEE.

Lesser, V., Ortiz, C., & Tambe, M. (Eds.). (2003). *Distributed Sensor Networks: A Multiagent Perspective*, Vol. 9. Kluwer Academic Publishers.

Maheswaran, R. T., Pearce, J. P., & Tambe, M. (2004a). Distributed algorithms for dcop: A graphical-game-based approach.. In Bader, D. A., & Khokhar, A. A. (Eds.), *ISCA PDCS*, pp. 432–439. ISCA.

Maheswaran, R. T., Tambe, M., Bowring, E., Pearce, J. P., & Varakantham, P. (2004b). Taking dcop to the real world: Efficient complete solutions for distributed multi-event scheduling.. In *AAMAS*, pp. 310–317. IEEE Computer Society.

Meier, M., Haschke, R., & Ritter, H. J. (2014). Perceptual grouping through competition in coupled oscillator networks. *Neurocomputing*, *141*, 76–83.

Meisels, A., Kaplansky, E., Razgon, I., & Zivan, R. (2002). Comparing performance of distributed constraints processing algorithms. In *AAMAS - Workshop on Distributed Constraint Reasoning DCR*, pp. 86–93. AAAI Press.

Modi, P. J., Shen, W.-M., Tambe, M., & Yokoo, M. (2005). Adopt: asynchronous distributed constraint optimization with quality guarantees.. *Artif. Intell.*, *161*(1-2), 149–180.

Modi, P. J., & Veloso, M. (2004). Multiagent meeting scheduling with rescheduling. In *In Distributed Constraint Reasoning, (DCR).*

Nemenyi, P. (1963). *Distribution-free multiple comparisons.* Ph.D. thesis, Princeton University, New Jersey, USA.

Nguyen, D. T., Yeoh, W., & Lau, H. C. (2013). Distributed gibbs: A memory-bounded sampling-based dcop algorithm. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS, pp. 167–174, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Okamoto, S., Zivan, R., & Nahon, A. (2016). Distributed breakout: Beyond satisfaction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pp. 447–453. AAAI Press.

Ottens, B., Dimitrakakis, C., & Faltings, B. (2017). Duct: An upper confidence bound approach to distributed constraint optimization problems. *ACM Trans. Intell. Syst. Technol.*, *8*(5), 69:1–69:27.

Pearce, J. P., Maheswaran, R. T., & Tambe, M. (2005). How local is that optimum? k-optimality for dcop. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, pp. 1303–1304, New York, NY, USA. ACM.

Pearce, J. P., Tambe, M., & Maheswaran, R. T. (2008). Solving multiagent networks using distributed constraint optimization.. *AI Magazine*, *29*(3), 47–62.

Pikovsky, A., Rosenblum, M. G., & Kurths, J. (2001). *Synchronization, A Universal Concept in Nonlinear Sciences*. Cambridge University Press, Cambridge.

Rust, P., Picard, G., & Ramparany, F. (2016). Using Message-passing DCOP Algorithms to Solve Energy-efficient Smart Environment Configuration Problems. In Kambhampati, S. (Ed.), *International Joint Conference on Artificial Intelligence*, pp. 468–474, New York, United States. AAAI Press.

Sherman, A., Rinzel, J., & Keizer, J. (1988). Emergence of organized bursting in clusters of pancreatic beta-cells by channel sharing.. *Biophys J*, *54*(3), 411–425.

Skardal, P. S., & Restrepo, J. G. (2012). Hierarchical synchrony of phase oscillators in modular networks. *Physical Review E*, *85*(1), 016208.

Strogatz, S. H. (1997). Spontaneous synchronization in nature. In *Frequency Control Symposium*, pp. 2–4. IEEE.

Strogatz, S. H. (2003). *Sync: The Emerging Science of Spontaneous Order*. Hyperion.

Vinyals, M., Shieh, E., Cerquides, J., Rodriguez-Aguilar, J. A., Yin, Z., Tambe, M., & Bowring, E. (2011). Quality guarantees for region optimal dcop algorithms. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '11, pp. 133–140, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications*, Vol. 8. Cambridge university press.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of śmall-worldńetworks. *Nature*, *393*(6684), 440–442.

Wolfram, S. (2003). *The Mathematica book* (Fifth edition). Wolfram Media, Champaign, IL, USA.

Wu, J., Jiao, L., Li, R., & Chen, W. (2011). Clustering dynamics of nonlinear oscillator network: Application to graph coloring problem. *Physica D: Nonlinear Phenomena*, *240*(24), 1972–1978.

Zhang, W., Wang, G., Xing, Z., & Wittenburg, L. (2005). Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks.. *Artif. Intell.*, *161*(1-2), 55–87.

Zivan, R., Okamoto, S., & Peled, H. (2014). Explorative anytime local search for distributed constraint optimization. *Artif. Intell.*, *212*(1), 1–26.