Advanced Approaches for Multi-Robot Coordination in Logistic Scenarios

Alessandro Farinelli[a], Nicolo' Boscolo[c], Elena Zanotto[b], Enrico Pagello[b]

[a]*Dept. of Computer Science, University of Verona, I-37134, Verona (Italy)*
[b]*Dept. of Electronics and Engineering, Via Gradenigo 6, I-35131 Padova (Italy)*
[c]*IT+Robotics Srl, Contrà Valmerlara 21, 36100 Vicenza (Italy)*

Abstract

Robotics technology has recently matured sufficiently to deploy autonomous robotic systems for daily use in several applications: from disaster response to environmental monitoring and logistics. In such applications, robots must establish collaborative interactions so to achieve their individual and collective goals and a key problem is for robots to make individual decisions so to optimize a system wide objective function. This problem is typically referred to as coordination. In this paper, we first describe modern optimization techniques for coordination in multiRobot systems. Specifically, we focus on approaches that are based on algorithms widely used to solve graphical models and constraint optimization problems, such as the max-sum algorithm. We then analyse the coordination problem faced by a set of robots operating in a warehouse logistic application. In this context robots must transport items from loading to unloading bays so to complete packages to be delivered to customers. Robots must cooperate to maximizes the number of packages completed in the unit of time. To this end a crucial component is to avoid interferences when moving in the environment. We show how such problem can be formalised as a Distributed Constrained Optimization problem and we provide a solution based on the binary max-sum algorithm. Finally, we provide a quantitative evaluation of our approach in a simulated scenario using standard robotics tools (ROS and Gazebo).

Keywords:
multirobot systems, coordination, task assignment, logistic

1. Introduction

Autonomous robotic systems are now becoming a crucial technology for several diverse application domains. Thanks to the increased reliability of platforms and sensors, as well as the reduced price of technology, robotic platforms are now used daily for several tasks such as domestic services (e.g., the Roomba vacuum cleaner) or industrial manufacturing. A key distinctive element for this new application domains is the high level of autonomy exhibited by such

systems which is in sharp contrast to robotic applications developed few years ago. Autonomy is now a crucial component for robotic technology and there are various initiatives to assess and challenge autonomous robotic systems in various scenarios. A recent example is the EU project RockIn[1] that aims at fostering the use of autonomous robots through competitions in two main areas: domestic services and industrial manufacturing.

Now, the increased use of robotic platforms and the high level of autonomy lead the way towards the deployments of Multi-Robot Systems (MRS), where several platforms operate in the same environments. A clear example of this is the Kiwa system [39], where several autonomous driving units transport materials in different areas of a warehouse.

When MRS must be deployed, a key component to ensure high efficiency for the system is the ability to orchestrate the different actions of the robots so to maximize the system performance. In other words, the MRS must be equipped with coordination techniques so that each robot will choose its single actions to maximize a system wide objective.

In this paper, we consider coordination approaches for MRS, with a specific focus on industrial application and specifically warehouse logistics. In particular, we consider a set of robots involved in transportation tasks in a warehouse, and we propose a coordination approach to maximize the task throughput (i.e., number of task completed in the unit of time).

Several previous approaches have considered the coordination problem for MRS, and nowadays there is a wide range of techniques that can be used to coordinate MRS [17, 9, 22]. One of the most well studied problems for coordination in MRS is *task assignment* [17, 22], where a set of robots must be assigned to a set of tasks to maximise a system-wide objective function (e.g., complete the maximum number of tasks, minimize the make-span, etc.). The task assignment problem has been tackled with various techniques such as Mixed Integer Programming (MIP) approaches [20], reactive methods [27] and biologically inspired approaches [24]. Arguably one of the most widely used approach for task assignment is the market based framework [8], where robots exchange valuations (i.e., bids) to execute tasks and the task is assigned to the robot with the best valuation.

Recent approaches to coordination investigate the use of graphical models and decentralised optimization approaches for MRS coordination [1, 4] and in particular for task assignment [13, 31]. Graphical models are appealing tools as they exploit the structure of the domain, that in MRS typically relates to locality of interactions: the actions of a robot typically affects only a small subset of other robots (e.g., robots in close proximity or robots assigned to close-by tasks). Graphical models provide extremely efficient techniques for optimization, and have been used in different community such as machine learning [3] and constraint processing [7]. Here we focus on a specific type of graphical model, i.e., the Distributed Constraint Optimization Problems (DCOPs) [25] as

[1] see `http://rockinrobotchallenge.eu/` for more details.

we are interested in decentralised solution.

In more detail, in this paper, we provide a DCOP model for the task assignment problem faced by robots involved in logistics operations in a warehouse. Among the various solution approaches for DCOPs we advocate the use of heuristic algorithms, and specifically the max-sum, an iterative message passing approach that has been shown to provide solutions of high quality for systems operating in real-time and with limited computation and communication resources [11]. The max-sum approach works very well when the problem can be well decomposed, i.e., when the decisions of one robot affect only a small subset of team-members, because the message update step, a key operation of max-sum, has a computational complexity that is exponential in the number of robots that can perform the same task. This exponential element can be a significant limitation for large scale, real-time systems. To combat this, recent approaches [35] show that for specific types of constraints and by using binary variables, such exponential element can be reduced to a polynomial. Hence, for models that involve only this type of constraints (called Tractable Higher Order Potentials or THOPs) and binary variables, we can use the max-sum approach (called Binary Max-Sum or BMS) for large-scale systems that must operate with real-time constraints. This important result has been exploited by relevant literature in different application domains such as UAVs coordination [31] and rescue operations [30]. Following this recent literature, here, we provide a binary DCOP model that involves only THOPs to encode our task assignment problem. A key feature of our model is the possibility to explicitly represent spatial constraints in the task assignment problem, hence minimizing harmful interference among the robots that severely limit the effectiveness of the system for warehouse logistic operations.

In summary, the main novelty of this work is the formalization of the coordination problem related to logistic scenarios as a DCOP, where we explicitly represent the interferences among the robots in the model. Moreover, we show that such model can be encoded using only THOPs, hence we can devise a tractable solution approach based on the max-sum algorithm.

We provide a quantitative evaluation of our approach in a simulated scenario using standard robotics tools (i.e., ROS and Gazebo). We compare the BMS approach to a greedy local search algorithm (i.e., DSA [15]), and our results show that the BMS provides better solutions by minimizing robots interferences.

The rest of the paper is organized as follows: Section 2 provides necessary background on MRS coordination approaches, focusing on graphical model, DCOPs and BMS. Section 3 details our model for task assignment in warehouse logistic operations, and Section 4 presents our empirical evaluation of the proposed approach. Finally, Section 5 concludes the paper.

2. Background and Related Work

In this Section, we detail the main issues for MRS coordination in industrial domains, then we provide a detailed discussion on coordination approaches for MRS, highlighting challenges and main solution techniques. Finally, we provide

necessary background on coordination approaches for multi-robot systems based on decentralized optimization and specifically on factor graphs.

2.1. MRS for Industrial Applications

The use of robots for industrial tasks in the broad area of manufacturing and logistics increased significantly over the last years, and several initiatives target industrial scenarios as an application domain. For example, the recent EU project RockIn considers the use of robots in industrial environments as one of the main challenges of the project. Specifically the RockIn project aims at pushing forward the innovation in cognitive and robotic systems through the use of competitions, and proposes two challenges: the *@home* challenge, which targets service robotics and the *@work* challenge which focuses on innovative industrial robots that perform various tasks such as pick-and-place and quality control. A crucial aspect of the project is the interest towards robotic systems that can closely interact with humans and become a sort of personal assistant of the workers. This requires robots to operate in a dynamic environment and to have a continuous interaction with human operators. Such operation modalities are in sharp contrast with respect to the standard use of robotic systems in the industry, where robots are typically isolated in highly engineered manufacturing cells and perform highly specialized tasks.

In this paper, we also focus on industrial scenarios where robots have a high degree of autonomy and operate in a dynamic environment. However, our focus is not on single robot capabilities (such as object manipulation or recognition) but rather on the use of a MRS where coordination plays a crucial role.

In this perspective, the Kiwa system [39] offers a radically innovative and interesting approach to material handling in warehouse management, by using a large team of autonomous driving units to transport storage pods in different areas of a warehouse. The main innovation behind the Kiwa system is the use of a coordinated approach where independent robots negotiate tasks to optimise the material handling operations.

In this work, we consider a similar setting where a set or robots are involved in transportations tasks for logistics. However, we focus on the specific problem of task assignment explicitly considering spatial constraints in the task allocation process. In particular, we consider the paths that robots execute to accomplish tasks and we aim at choosing task assignments that minimize spatial interferences between different robots. Moreover, we propose the use of advanced techniques for coordination that are based on graphical models and DCOPs solutions. Specifically, we devise a binary THOP-only factor graph model and investigate the Binary Max-Sum algorithm as a solution approach.

2.2. Coordination in MRS

Coordination for MRS has been investigated from several diverse perspectives and nowadays, there is a wide range of techniques that can be used to orchestrate the actions and movements of robots operating in the same environment. Specifically, the ability to effectively coordinate the actions of a MRS is a key

requirement in several application domains that range from disaster response to environmental monitoring, military operations, manufacturing and logistics. In all such domains, coordination has been addressed using various frameworks and techniques and there are several survey papers dedicated to categorize such different approaches and identifying most prominent issues when developing MRS [5, 37, 9].

Given our focus on logistic scenarios, here we restrict our attention to coordination approaches based on optimization and specifically on task assignment [17, 22] as this is the most common framework for our reference application domain [39]. Overall, the problem of task assignment can be roughly stated as the problem of deciding which robot of the MRS performs which task(s). This decision process has a great impact on the performance of the overall system and therefore has been deeply studied and investigated. In particular, in [17] Gerkey and Mataric propose a taxonomy to analyze the different approaches to the problem of Multi Robot Task Allocation (MRTA), along with a formal framework for the study of this problem. The authors consider three main dimensions: 1) single-task (ST) vs multi-task (MT) robots, based on whether the robots involved in the task assignment process can execute more than one task at a time; 2) single-robot (SR) vs multi-robot (MR) tasks, considering whether the tasks to be performed involve one or multiple robots; 3) instantaneous assignment (IA) vs. time-extended assignment (TA), distinguishing whether the information concerning the robots, tasks and environment permit only an instantaneous assignment or a more sophisticated planning approach. The authors provide a formal characterization of a wide set of MRTA problems, analyzing and classifying in their taxonomy significant approaches used in MRS literature; for those approaches, bounds to the optimality of the method used with respect to the optimal solution are provided. Recently, Korsah and colleagues [22] present a new taxonomy that progress beyond the work by Gerkey and Mataric by explicitly taking into account the issues of interrelated utilities and constraints across task execution, i.e., tasks are not independent and executing a task has effect on the reward/cost of other tasks in the system.

Coordination based on task assignment has been addressed from several perspectives ranging from Mixed Integer Programming (MIP) approaches [20] to decentralized reactive methods [27] and biologically inspired techniques [24]. In particular, market based approaches represent a widely used framework for task allocation in robotic systems in various application domains such as exploration and mapping [42] and multi-robot patrolling [29] (see [8] for a recent survey). In such framework robots use auction-based mechanisms to negotiate tasks and reach a consensus on the allocation. The basic idea is that robots exchange "bids" with their team-mates, and such bids encode how well each robot can perform a given task. The bid computation is typically domain dependent as it must consider application specific constraints/features. The use of different auction mechanisms and their effect on task allocation has been deeply studied in the relevant literature and the main approaches can be broadly divided in three main categories: i) *combinatorial auctions*, where robots bid for set of tasks, and hence they can consider synergies and constraints that might hold

among such tasks [2]. ii) *repeated, parallel single item auctions*, where robots bid on a single task in parallel and the auctions are repeated periodically. This scheme does not consider synergies among tasks, hence, as stated in [19], the overall performance of the system can be arbitrarily far from the optimal. However, unlike combinatorial auctions, parallel single item auctions are very fast to solve and robust to possible unexpected problems [34]. iii) *sequential single-item auctions*, where robots bid on single tasks, but such tasks are auctioned sequentially and when a robot bids on a task it considers synergies with all tasks it is currently responsible for. This type of auctions has been frequently used as it represents a good compromise between computation requirement and quality of solution. In particular, in specific settings, this scheme can provide quality guarantees on the solution (see [36] for more detail). Often, such auction schemes are hybridized to produce solutions that are both effective and robust to possible unexpected situations [26].

Another framework that has been widely used for task assignment is the one based on token passing originally proposed by Scerri and colleagues [32]. In this context *tokens* represent tasks to be executed and are exchanged through the system in order to collect information and to allocate the tasks to the robots. When a robot receives a token, it decides whether to perform the task associated to it or to pass the token on to another robot. This decision is taken based only on local information: each robot follows a greedy policy, i.e. it tries to maximize its utility, given the tokens it can currently access, its resource constraints and a broad knowledge on team composition. However, tasks are executed by the robot that has the corresponding token only if its capability is higher than a given threshold. Such thresholds guide the search towards good solutions for the allocation problem, and while such mechanism cannot provide guarantees concerning the optimality of the solutions found, in practice it consistently increases the algorithm performance [32]. The main benefit of the token passing scheme is the extremely low computation and communication requirements that makes such approach very well suited for robotic scenarios [10, 40], large scale applications [33] and situations where communication can be severely limited (e.g., application scenarios such as incident management and rescue, where bandwidth is limited and the number of messages must be minimized) [14] .

Following recent results in MRS literature [1, 13, 31], in this paper we investigate a different perspective to task assignment and coordination, by using decentralized optimization techniques and proposing a representation of the coordination problem based on graphical models. In the next Section, we describe the necessary background on graphical models, Distributed Constraint Optimization Problems and the most prominent solution techniques (i.e., Distributed Stochastic Algorithm and the max-sum).

2.3. Graphical Models for Coordination

The main idea behind the use graphical models for coordination is to exploit the structure of the domain to provide efficient techniques to solve the problem. For example, consider our warehouse logistic scenario, while there could potentially be several orders to consider and several robots to control, the actions of one

such robot typically impact only on the future decisions of a subset of its teammembers. Graphical models represent a general and powerful framework to encode such structure and to exploit the possible loose dependency of different elements in the system so to alleviate computation. In particular, graphical models have been used in different research communities with different names: e.g., probabilistic graphical models [3], cost networks [7], DCOPs [25]. However they all share similar key concepts and solution techniques [18]. Here we focus on DCOPs as we are interested in decentralized solutions.

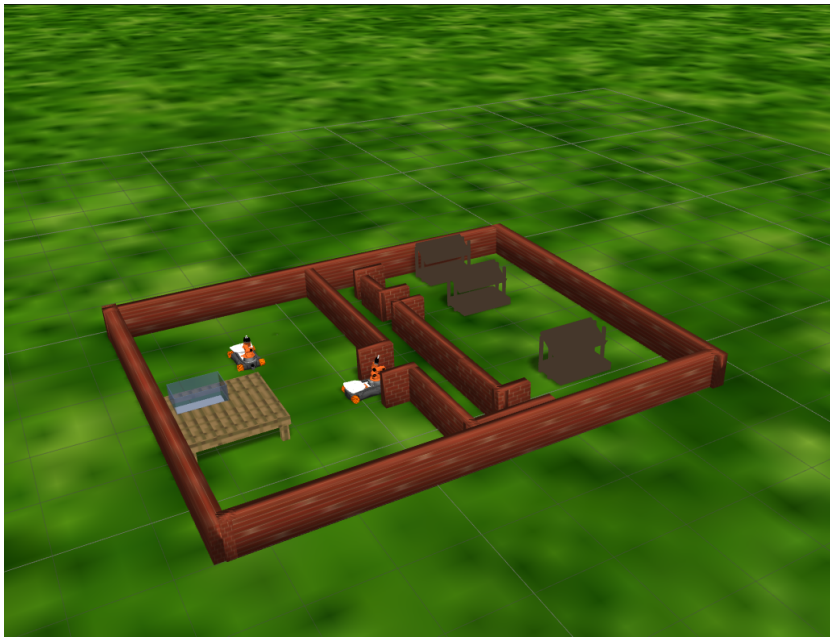

Figure 1: An exemplar situation for MRS coordination in warehouse logistic (inspired by the RockIn@Work challenge). Robots must transport items from the three loading bays (right part of the image) to the unloading bay (left).

To give an intuition of how we can model our application scenario by using DCOPs, Figure 1 shows an exemplar configuration of robots and tasks in a simple warehouse scenario. Tasks in this context require to pick up, load, and transport various items from three loading bays (visible in the right side of the picture) to the unloading bay (left). Robots should avoid accessing the same loading bay, moreover they should try to avoid as much as possible interference in the corridor connecting the bays. Considering this, a possible cost network representing this situation is reported in Figure 2. This diagram uses a factor graph [23] to represent the cost network, where circles represent variables and squares represent factors. A factor is associated to a cost (or utility) that provides a value for each possible configuration of the variables linked to that factor.

Variables in this case represent possible allocations of robots to tasks (i.e., loading bays), hence the domain for variable r_1 and r_2 will be $D_1 = \{t_1, t_2, t_3\}$. Our objective, is then to find the assignment of variables that maximises the sum of the functions (in this paper we consider the function as utilities hence we maximise their sum).

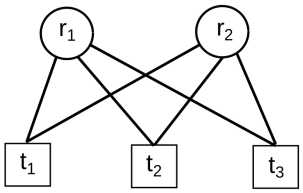

Figure 2: A factor graph representation of the scenario reported in Figure 1.

More in general a DCOP for a multi-robot coordination problem is a tuple $\langle \mathcal{R}, \mathcal{X}, \mathcal{D}, \mathcal{F} \rangle$, where $\mathcal{R} = \{r_1, \ldots, r_m\}$ is a set of robots and $\mathcal{X} = \{x_1, \ldots, x_s\}$ is a set of variables, where each variable x_i is owned by exactly one robot r_i, but a robot can potentially own more than one variable. The robot r_i is responsible for assigning values to the variables it owns. $\mathcal{D} = \{D_1, \cdots, D_s\}$ is a set of discrete and finite variable domains, and each variable x_i can take values in the domain D_i. Then, $\mathcal{F} = \{f_1, \ldots, f_n\}$ is a set of functions that describe the constraints among variables. Each function $f_i : D_{i_1} \times \cdots \times D_{i_{k_i}} \rightarrow \Re \cup \{-\infty\}$ depends on a set of variables $\mathbf{x_i} \subseteq \mathcal{X}$, where $k_i = |\mathbf{x_i}|$ is the arity of the function and $-\infty$ is used to represent hard constraints. Each function assigns a real value to each possible assignment of the variables it depends on.

The goal is then to find a variable assignment that maximises the sum of constraints:

$$\arg\max_{\mathbf{x}} \sum_{i} f_i(\mathbf{x_i}) \tag{1}$$

To date, there are a significant number of solution techniques for DCOPs, and they can be broadly divided into two main families: exact algorithms that are guaranteed to provide the optimal solution (such as ADOPT [25] and DPOP [28]) and heuristic approaches such as greedy local techniques (e.g., DSA [41]) or max-sum [12]. Now, while exact algorithms find useful application within large computational systems, they do not address many of the additional challenges that are present when considering robotic systems that must operate in dynamic environment. In particular, to guarantee optimality of solution all the above exact algorithms demand that some aspects of the algorithm grows exponentially in size (because finding an optimal solution for a DCOP is an NP-hard problem [25]) and such exponential element can not be accepted in our reference

application domain. In contrast, heuristic approaches can not guarantee optimal solutions but they do provide a viable alternative for robotic applications as they require moderate computation and communication resources.

Among heuristic approaches, DSA is one of the most prominent technique. In short, DSA is a distributed local greedy search technique, which uses a stochastic element to mitigate possible misalignments in robots' assignments. In more detail, when using DSA each robot initializes its variables with a random assignment and then they perform a series of *local* moves to optimize the objective function. A *local move* involves changing the joint assignment of the variables corresponding to a subset of the robots (aka *neighborhood*) to optimize the local *gain* (the difference between the sum of local utility functions evaluated with the new assignment with respect to the previous one). In a decentralized context robots evaluate and perform local moves in parallel, informing their neighbours of the new assignment after each move. Such parallel execution without coordination can result in poor system performance since robots can act based on out-of-date knowledge about the choices of other robots. To combat this, DSA introduces stochasticity by only allowing each robot to optimize its local gain if a random number exceeds a certain activation probability. Such activation probability must be empirically tuned for the application domain. Overall, DSA provides a low cost heuristic solution to DCOPs and it has been used in various contexts [15], however due to the greedy nature of the optimization it can often return solutions of poor quality.

The max-sum algorithm has been frequently used in the probabilistic graphical model community [3, 16] and offers a very different perspective for solving DCOPs. In such approach, each robots tries to compute an estimation of the impact that each of its action has on the global optimization function. Such an estimate (usually called the belief function) is build up by iteratively exchanging messages with neighbours, and once this belief function is built each robot chooses the assignment that maximizes this function. The max-sum operations are best understood considering the associated factor graph, in more detail, at each execution step, each robot computes and sends two types of messages: the variable to function ($q_{x_i \to f}(x_i)$) message and function to variable ($r_{f_V \to x_j}(x_j)$) messages. The equations to compute such messages are reported below[2]:

Variable to function messages:

$$q_{x_i \to f}(x_i) = \sum_{g \in \mathcal{M}_i, g \neq f} r_{g \to x_i}(x_i) \tag{2}$$

where $\mathcal{M}_i$ represents the set of funtions connected to x_i.

Function to variable messages:

$$r_{f_V \to x_j}(d) = \max_{d_{k_1}, \cdots, d_{k_s}} [f_V(d, d_{k_1}, \cdots, d_{k_l}) + \sum_{k \in (k_1, \cdots, k_l)} q_{x_k \to f_V}(d_k)] \tag{3}$$

[2]We refer the interest reader to [6] for a more detailed description of the algorithm.

were, $(d_j, d_{k_1}, \cdots, d_{k_l})$ represents the joint variable assignments of variables $V = (x_j, x_{k_1}, \cdots, x_{k_l})$, and d is one possible value of D_j. Notice that the message update step for a function that is connected to l variables requires in general exponential operations in l (i.e., $\mathrm{O}(|D|^l)$ where $|D|$ is the cardinality of the largest domain among the l variables). This can represent a computational bottleneck depending on the specific application scenario.

At any time during the propagation of these messages, robot r_i is able to determine which value it should adopt such that the sum over all the robots' utilities is maximised. This is done by locally calculating the belief function, $z_i(x_i)$, from the messages flowing into robot i's variable node:

$$z_i(x_i) = \sum_{g \in \mathcal{M}_i} r_{g \to x}(x) \tag{4}$$

and then finding $\arg\max_{x_i} z_i(x_i)$

The max-sum iterative message passing process terminates when the messages reach a fixed point (i.e., new messages are identical to previous messages). However, the max-sum algorithm is guaranteed to converge to a fixed point only if the factor graph is acyclic and in this case it computes the optimal assignment. In more general settings, there are only limited guarantees on convergence and solution quality [38], however extensive empirical results [12, 21] show that when executed on a loopy factor graph max-sum often achieves very good solutions. Since in these cases convergence is not guaranteed the iteration process is usually performed for an arbitrary (relatively small) number of coordination cycles.

The max-sum algorithm has been recently used in various application domains and specifically for coordination of multi-robot systems [13, 1, 4]. However, as mentioned above the computational complexity associated to the message update step for function-to-variable messages can be a problem when there are factors that depends on several variables or when the domain of the variable is large. In the next Section we describe a recent, promising direction to combat this bottleneck.

2.4. Binary Max-sum and Tractable Higher Order Potentials

As mentioned above the max-sum algorithm is a widely used tool in the probabilistic graphical model community. In such community, the computational bottleneck for the message update phase has been recently addressed by Tarlow and colleagues [35] which have shown that for specific types of constraints such message update can be reduced to polynomial time (between $O(k)$ and $O(k \log(k))$), depending on the constraint types, where k is the number of variables connected to the constraint. These constraints are called Tractable Higher Order Potentials (THOPs) and authors have shown that by exploiting the specific structure of these constraints it is possible to update messages while avoiding the complete enumeration of all variables' configurations. Tarlow and colleagues identify several types of THOPs, however the most important ones, and the ones that we will use here, are the so called *cardinality* potentials. The

distinctive feature of these potentials is that the value of the associated function depends on the number of variables that are active (i.e., the number of variables that take value 1), but not on which specific variable is active. In general, a cardinality potential can be expressed as $f(\mathbf{X}) = f'(n_a(\mathbf{X}))$, where $\mathbf{X}$ represents one possible configuration of the variables and $n_a(\mathbf{X}) = \sum_{x_i \in \mathbf{X}} \mathbf{x_i}$, i.e., the number of active variables in the configuration encoded by $\mathbf{X}$. Tarlow and colleagues have shown that for cardinality potentials the complexity of the message update phase for function-to-variable messages (i.e., equation 3) can be reduced to $O(k \log(k))$. Particular cases of cardinality potentials are strict cardinality potentials such as *OneAndOnlyOne*, *AtMostOne* and *AllZeros*. For some of these strict cardinality potentials it is possible to devise even more efficient procedures to update the messages, and in particular Pujol-Gonzales and colleagues in [31] show that for *OneAndOnlyOne* potentials the messages can be computed in $O(k)$.

As mentioned, such reduced complexity is possible only for constraints that exploit a specific structure, however, previous work has shown that for many application domains such potentials are expressive enough to encode the related optimization problem [31, 30]. Specifically, cardinality potentials are very frequently used for encoding coordination problems. For example in a task assignment problem one typically must express that at most one robot is assigned to one task, this can be easily represented as a function that returns 0 for all configuration where at most one variable is active, and $-\infty$ otherwise (assuming a maximization problem).

Typically THOPs are defined over binary variables hence the associated solution technique is called Binary Max-Sum. The main idea to represent a task assignment problem with a binary model (i.e., a model where the domain of every variable can take only 2 values) is to use *decision* variables to represent the fact that a specific robot is allocated to a specific task. In more detail, if a variable r_i has a domain $D_i = t_1, \cdots, t_k$ (i.e., robot r_i can be assigned to k tasks), in the binary model we have k decision variables $< a_{i,1}, \cdots, a_{i,j}, \cdots, a_{i,k} >$ where $a_{i,j} = 1$ represents the fact that robot i is allocated to task j and $a_{i,j} = 0$ means that the robot is not allocated to the task. Moreover, if a robot can execute only one task we must also create a k-ary constraint $S_i(a_{i,1}, \cdots, a_{i,j}, \cdots, a_{i,k})$ that ensure that only one variable will be active for each robot (i.e., $\forall i; \sum_j a_{i,j} \leq 1$). To better see this, Figure 3 reports the binary versions of the factor graph in Figure 2. The diagram shows the binary variables that represent robots' decisions and the factors that ensure one robot will only execute one task at a time. For example variables $\{x_{21}, x_{22}, x_{23}\}$ represent the possible allocations for robot r_2 and the factor S_2 ensures that only one of those variables will be active. Moreover, for each task (e.g., t_1) we have one factor that encodes the value (or penalty) for having more than one robot allocated to that task (e.g., C_1) and one factor for each robot that encodes how well each robot can perform that task in isolation (e.g., u_{12} and u_{21}). These last factors are unary factors as they depend only on one variable, and take into account all elements that are specific for a task and a robot, such as the distance from the robot to the task, robot's capabilities related to the task (e.g., whether the robot can perform the

actions required by the task) and so forth.

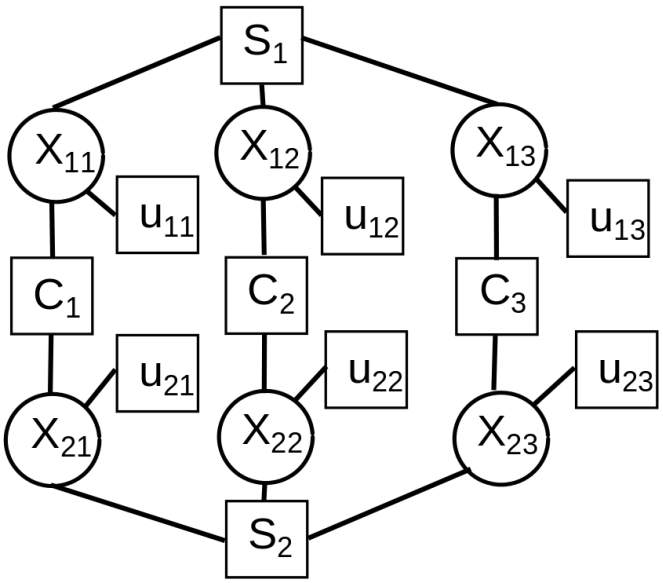

Figure 3: The binarized version of the factor graph reported in Figure 2.

Summarizing, to have a tractable message update phase for the max-sum approach we can create a model that is composed of binary variables and constraints that are THOPs. In Section 3 we provide such a model for our specific application scenario.

3. A Binary Max-Sum Approach for MRS Coordination

In this Section we detail our reference scenario for MRS coordination and then we provide a binary THOP model of such scenario.

3.1. Problem Description

Our reference scenario is based on a warehouse that stores items of various types. Such items must be composed together to satisfy orders that arrive based on customers' demand. For example, think of a wholesale bookseller that sells books to bookshops. To reduce transportation costs associated to the book delivery the wholesale bookseller waits for several single-book orders to queue up and packages the books into one case that is then shipped to the bookseller. Hence, in our reference scenario we have items of various types that are stored in particular sections of the building (i.e., *loading bays*) and must be transported to a set of *unloading bays* where such items are then packed together by human operators. The set of items to be transported and where they should

go depends on the orders. Here we do not consider the optimization process associated to which of the orders that the wholesaler receives should be passed on to the warehouse transportation system, instead we consider only the problem of serving the orders once they have been passed on to one of the unloading bays. Hence, in our domain, a set of robots is responsible for transporting items from the loading bays to the unloading bays and the system goal is to maximize the throughput of the orders, i.e., to maximize the number of orders completed in the unit of time. Now, robots involved in transportation tasks move around the warehouse and are likely to interfere when they move in close proximity, and this can become a major source of inefficiency (e.g., robots must slow down and they might even collide causing serious delays in the system). Hence, a crucial aspect to maintain highly efficient and safe operations is to minimize the possible spatial interferences between robots. Specifically, here we propose to take this interferences into account in the task assignment process and assign tasks to robots so to reduce the possible interferences among the transportation robots.

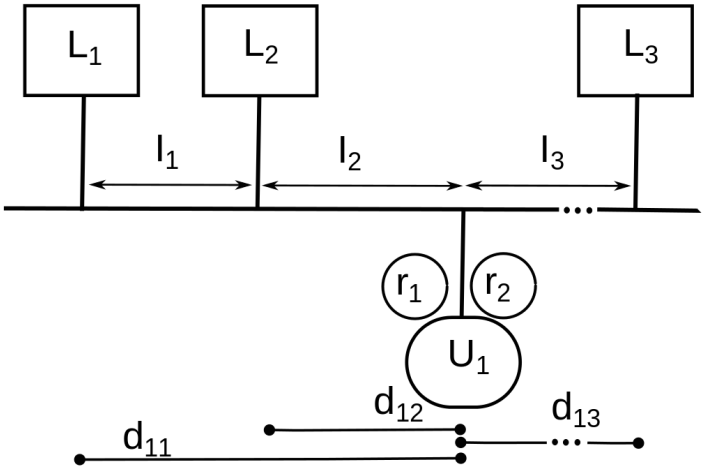

Figure 4: A schema of an exemplar scenario with 3 loading bays $\{L_1, L_2, L_3\}$, 2 unloading bays $\{U_1\}$ and 2 robots $\{r_1, r_2\}$. The line segments $\{I_1, I_2, I_3\}$ represent the routes that connect the loading and unloading bays. We assume that the distance $d_{1,3}$ between U_1 and L_3 is greater than $d_{1,1}$ and $d_{1,2}$.

Figure 4 reports a diagram depicting a schema of an exemplar scenario for our warehouse coordination problem. The scenario includes, three loading bays $\{L_1, L_2, L_3\}$, one unloading bay $\{U_1\}$ and two robots $\{r_1, r_2\}$ positioned in U_1. The line connecting the bays represent the optimal route that robots should follow to transport items, and we indicate relevant segments of this route with $\{I_1, I_2, I_3\}$. When robots move across the bays they will go through some of

this segments, for example if robot r_1 must go from U_1 to L_1 it will go through segments $\{I_1, I_2\}$. We consider a spatial interference when two robots must traverse the same segments to complete their tasks, hence our goal is to assign tasks to robots while minimizing such spatial interferences. For example, assume we have three tasks that involve transporting one item from each of the loading bays to the unloading bay: $\{< L_1 \rightarrow U_1 >, < L_2 \rightarrow U_1 >, < L_3 \rightarrow U_1 >\}$ and that both robots can execute all such tasks. Moreover, we assume that the travel distance from U_1 to L_3 is greater than both the distances to L_1 and L_2. In this case, if we do not consider the spatial interferences, one of the best allocations (i.e., the ones that minimize travel time for all robots and consequently the task throughput) will assign r_1 to $L_1 \rightarrow U_1$ and r_2 to $L_2 \rightarrow U_1$,[3], however this will create a spatial interference on segment I_2 that might significantly slow down the actual task completion time. On the other hand assigning r_1 to L_2 and r_2 to L_3 will result in a longer travel time but it will avoid any spatial interference.

In what follows we provide a binary THOP model for this task assignment problem that explicitly takes spatial interferences into account.

3.2. Formalization

We formalize the MRS coordination problem described above as a task allocation problem where robots must be allocated to transportation tasks. In our formalization we impose that robots can execute only one transportation task at a time, hence we do not explicitly represent sequence of tasks for the single robot, moreover, if transportation tasks are more than the available robots (as it is usually the case in most practical scenario) at each time step only a subset of the tasks will be allocated. However, since the task allocation process is repeated over time robots effectively serve a sequence of tasks. The rationale behind this choice is twofold: first, since we assume no knowledge on future orders for the warehouse we prefer to avoid planning ahead of time as the situation may drastically change, e.g., a new order arrives that should be served first. Second, we aim at solving the allocation problem in a small fraction of time so to operate in real-time, hence we prefer to avoid reasoning over the sequence of task allocation. Moreover, we impose that one transportation task can be executed by only one robot, this is to avoid interferences for the loading and unloading operations. Following the taxonomy proposed by Gerkey and Mataric in [17], this is a single-task, single-robot, instantaneous-assignment problem. However, as mentioned above, we explicitly consider the spatial interferences among robots that move in the warehouse, hence we try to find an assignment that maximizes the task throughput considering possible delays due to such spatial interferences.

In more detail, our model considers a set of items of different types $E = \{e_1, \cdots, e_N\}$, stored in a set of loading bays (L_i). Each loading bay contains items of the same type. The warehouse must serve a set of orders $O =$

[3]Here we consider only the instantaneous allocation problem, the allocation process will then be repeated for the remaining tasks.

$\{o_1, \cdots, o_M\}$. Orders are processed in one of the unloading bays (U_j) one at a time. Each order is defined by a vector of demand for each item type (i.e., the number of required items to close the order). Hence, $o_j = \langle d_{1,j}, \cdots, d_{N,j} \rangle$, where $d_{i,j}$ is the demand for order j of items of type i. When an order is finished (i.e., all required items have been unloaded to the bay) a new one arrives, and, as mentioned before, we assume to have no knowledge on future orders.

The orders induce a set of $N \times M$ transportation tasks $T = \{t_{i,j}\}$, with $t_{i,j} = \langle d_{i,j}, P_{i,j}, pr_{i,j} \rangle$, where $t_{i,j}$ defines the task of transporting $d_{i,j}$ items of type i (hence from loading bay L_i) for order o_j (hence to unloading bay U_j). Each task has a set of edges $P_{i,j}$ that form the shortest path going from L_i to U_j. $pr_{i,j}$ is the priority for the task that is provided as an input to the transportation system, this can be related to the urgency of the order or to specific commercial agreements between the wholesaler and the order recipients[4].

We have a set of robots $R = \{r_1, \cdots, r_K\}$ that can execute transportation tasks, where each robot has a defined load capacity for each item type $C_k = \langle c_{1,k}, \cdots, c_{N,k} \rangle$, hence $c_{i,k}$ is the load capacity of robot k for items of type i.

The allocation of robots to transportation tasks is encoded by an allocation matrix $A = \{a_{i,j,k}\}$ where $a_{i,j,k} = 1$ if robot r_k is allocated to transportation task $t_{i,j}$ and 0 otherwise. Hence we have a total of $K \times N \times M$ binary variables. For each robot and each transportation task we define $\delta_{i,j,k}$ as a measure of travel time required to perform task $t_{i,j}$ given current position for robot r_k (without considering possible interferences with other robots).

Notice that, when we allocate a robot k to a task $t_{i,j}$ we do not impose any constraint on $c_{i,k}$ and $d_{i,j}$. This means that a robot k might be allocated to a task $t_{i,j}$ even if the robot can not transport all the demand for that task in a single trip (i.e., $c_{i,k} < d_{i,j}$). In this case, we load on the robot k the maximum amount of item units it is able to transport (i.e., $c_{i,k}$). Then at the next task assignment iteration we update the demand for the transportation task $t_{i,j}$ accordingly (i.e., the new demand for the transportation task is $d_{i,j} - c_{i,k}$). In this way even tasks that require a higher demand than the capacity of the robots will be accomplished in successive travels. Similarly, when $c_{i,k} > d_{i,j}$ the amount of units we can transfer to the destination is $min(c_{i,k}, d_{i,j})$. This is reflected in equation 5 that defines the value of assigning a robot k to a task $t_{i,j}$ (considering also the priority for the task and the travel time without interferences).

$$v_{i,j,k} = min(c_{i,k}, d_{i,j})(pr_{i,j} - \delta_{i,j,k}) \tag{5}$$

To represent spatial interferences among robots, we consider a set of transportation line segments $I = \{I_1, \cdots, I_Q\}$, and a mapping $\mathcal{M}$ from line segment to all pairs of transportation tasks and robots. Such mapping relates a line segment with all the pairs $< t_{i,j}, r_k >$ that use such line segment, i.e., $\mathcal{M} : I_q \mapsto \{t_{i,j}, r_k\}_q \subseteq T \times R$. A pair $< t_{i,j}, r_k >$ uses a line segment I_q if

[4]While we set a uniform priority in the experiments of Section 4, we maintain this concept in the model to show that our approach can incorporate such information.

the shortest path that robot r_k must perform to go from its current position to the loading bay and then to the unloading bay uses the line segment I_q. The interaction among robots happens on the line segments, where more robots on the same line segment generate an interference, hence slowing down the operations. Such interferences, are defined by a function $I_q(\mathbf{a}_q)$ that takes as input all allocation variables related to line segment q, and penalizes the presence of more robots on the line. Hence the function $I_q(\mathbf{a}_q) = f(\sum_q a_q)$ where with a slight abuse of notation we indicate with q the indices of the pairs <task,robot> related to line segment I_q as returned by the mapping $\mathcal{M}$ previously defined. Specifically, we consider as $f(\sum_q a_q) = \alpha(\sum_q a_q)^\eta$, with $\alpha \geq 0$ and $\eta \geq 0$.

Given this our objective function is:

$$\max_A \sum_{i,j,k} v_{i,j,k} a_{i,j,k} - \sum_q I_q(\mathbf{a}_q) \tag{6}$$

subject to:

$$\forall k \sum_{i,j} a_{i,j,k} \leq 1$$

i.e., one robot can do at most one task.

$$\forall (i,j) \sum_k a_{i,j,k} \leq 1$$

i.e., one task can be allocated to at most one robot.

Such constraints can be encoded in the objective function by including *selectors*, which are functions that return zero for valid configurations and ∞ otherwise. In more detail, we have robot selector functions R_k^s representing the constraint that each robot can execute at most one task:

$$R_k^s(\{a_{i,j,k}\}) = \begin{cases} 0 & \text{if } \sum_{(i,j)} a_{i,j,k} \leq 1 \\ \infty & \text{otherwise} \end{cases}$$

where variables $\{a_{i,j,k}\}$ are all decision variables that represent possible allocations for robot r_k.

Moreover, we have task selector functions $T_{i,j}^s$ representing the constraint that each task can be executed by at most one robot:

$$T_{i,j}^s(\{a_{i,j,k}\}) = \begin{cases} 0 & \text{if } \sum_k a_{i,j,k} \leq 1 \\ \infty & \text{otherwise} \end{cases}$$

where variables $\{a_{i,j,k}\}$ are all decision variables that represent possible robot allocations for task $T_{i,j}$.

Hence the complete objective function is:

$$\max_A \sum_{i,j,k} v_{i,j,k} a_{i,j,k} - \sum_q I_q(\mathbf{a}_q) - \sum_k R_k^s(\{a_{i,j,k}\}) - \sum_{i,j} T_{i,j}^s(\{a_{i,j,k}\}) \tag{7}$$

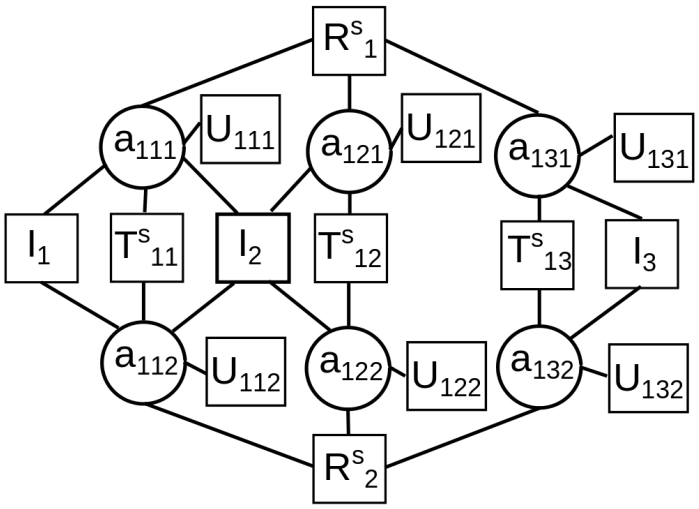

Figure 5: A binary factor graph model for the scenario represented in Figure 4.

Figure 5 reports a binary factor graph that encodes such objective function for the exemplar scenario represented in Figure 4. The factors R_1^s and R_2^s represent the selector factors for the two robots and the factors $\{T_{1,1}^s, T_{1,2}^s, T_{1,3}^s\}$ represent the selector factors for tasks. The factors $\{I_1, I_2, I_3\}$ are associated to the line segments and represent the possible spatial interferences among the robots. Notice that I_1 is linked only to variables $\{a_{1,1,1}, a_{1,1,2}\}$ because such segment will be used only by all robots associated to task $T_{1,1}$ (i.e., to go from the current unloading bay U_1 to the loading bay L_1 and back). Similarly I_3 is associated to variables $\{a_{1,3,1}, a_{1,3,2}\}$. However, I_2 is linked to variables $\{a_{1,1,1}, a_{1,1,2}, a_{1,2,1}, a_{1,2,2}\}$ because such segment will be used by all robots associated to task $T_{1,1}$ and task $T_{1,2}$. The connections between the I factors and associated variables can be easily devised by knowing the set of edges $P_{i,j}$ that are associated to each task. Finally, each variable is associated to a unary factor that encodes task-robot specific features. In our case the unary factor encodes the function $U_{i,j,k}(a_{i,j,k})$, where $U_{i,j,k}(1) = v_{i,j,k}$ and $U_{i,j,k}(0) = 0$.

We solve this model by using the max-sum algorithm where the message update phase exploits the structure of the higher order potentials as described in Section 2.4. In particular, the model involves one selector factor for each transportation task, one selector factor for each robot, one selector factor for each line segment and one unary factor for each variable. Hence if we have S transportation tasks, K robots and Q line segments we have $S \times K$ variables and a total of $S \times K + S + K + Q$ factors. For unary factors, the message derivation is not a problem as they will always send the same value (i.e., the factor $U_{i,j,k}$ will always send the value $v_{i,j,k}$). All not-unary factors are *cardinality potentials* hence the complexity of the message update phase is linearithmic

(i.e., $O(k \log(k))$). Hence the message update phase is extremely efficient and it is possible to handle extremely large factor graphs (i.e., comprising thousands of variables/factors see [16] for example).

The binary factor graph depends on the transportation tasks that are present in the system, hence it can change each time a task is completed. However, in this work we wait for all robots to finish their current tasks before building a new factor graph. The new factor graph is built by removing all tasks that have been executed, by considering all new tasks that may have appeared in the system and by updating the demand for all tasks that have been partially executed. Such factor graph is then solved by running BMS for a fixed amount of iterations and the decision is implemented by the robotic platforms that execute the tasks they are assigned to. This more conservative approach avoids task re-allocation that might be problematic in a logistic scenario where picking up and placing objects might require human intervention or might be difficult to be accomplished by the robots.

In the next session we describe our empirical evaluation of our approach.

4. Empirical Evaluation

In this Section we present the empirical evaluation of our approach. The goal of our empirical evaluation is: i) to validate the use of our DCOP model for task assignment in our logistic scenario; ii) to evaluate the performance of our BMS approach as opposed to other heuristic techniques for DCOPs. As mentioned in Section 2.3, providing the optimal solution for a DCOP is an NP-hard problem. Hence, for most practical settings it will be not feasible to compute an optimal allocation. While we could compute the optimal solution for small instances (i.e., few transportation tasks, few robots) this would not provide significant information on the quality of solutions returned by BMS for realistic scenarios. To this end we compare our BMS approach against DSA which is a standard greedy heuristic for solving DCOPs. Section 4.1 describes our implementation and the empirical methodology in more detail, while Section 4.2 presents and discusses the achieved results.

4.1. Implementation and Empirical Methodology

The BMS approach has been implemented using C++ language, the ROS framework and the Gazebo simulator for the scenario. The ROS architecture is intrinsically distributed therefore it is an ideal candidate for testing our decentralized coordination approach. In particular, the factor-graph nodes should be distributed over the agents. We define two kinds of agents which will be responsible for the message computation:

1. robot agents. They are in charge of the messages related to i) the variables that define the possible allocations of the robot; ii) the unary value factors; iii) the robot selector factor. For example, considering Figure 5 the robot agent for robot r_1 will be in charge of the message update for the variables $\{a_{1,1,1}, a_{1,2,1}, a_{1,3,1}\}$, for the unary factors $\{U_{1,1,1}, U_{1,2,1}, U_{1,3,1}\}$ and for

the selector factor R_1^s. This agent runs on the computational node that is in charge of controlling the robot.

2. supervisor agent. This agent is in charge of those factors "shared" among more robots such as those representing the spatial interferences and the tasks mutual activation. In our implementation we decide to have this agent running on the robot that has the lowest id among the ones that share each factor. For example, considering Figure 5, the supervisor agent will be in charge of all the three I_q factors and the three T_{ij}^s factors, and it will be hosted by the computational node for robot r_1. This choice does not have any impact on the execution of the algorithm but only on the computation and communication load for the computational node. This has never been an issue in our experiments.

The robot capacity for each kind of product (i.e., the C_k vector defined in Section 3.2) is chosen randomly over a standard normal distribution. The agents exchange all the messages over a local network hosted in a pc with standard features: 8 core Intel i7, 4GB RAM, 10MB cache. In our experiments, we fixed the number of total iterations for BMS to 10, according to our results this always allows the algorithm to converge to a valid solution (in the variable assignment).

The effectiveness of the BMS approach has been evaluated comparing with the DSA approach [15]. The DSA activation probability threshold has been empirically tuned over the Scenario 2 described in Table 1 and fixed to $p_a = 0.7$.

Our warehouse logistic scenario is defined by the number of loading and unloading bays, the number of robots, the topology of the environment and the orders that the transportation system must serve. The number of loading/unloading bays and the number of robots is an input that defines the specific scenario. As for the orders, a random order generator creates a set of loading bay requests for each unloading bay (the number of items to be collected from that bay is chosen randomly between 0 and 4 items). Hence for each scenario the number of orders depends on the number of loading and unloading bays. Moreover, depending on the demand and on the robot capacity, the number of orders induces a number of transportation tasks that are executed by the robots. On average in our experiments the number of transportation tasks for one scenario varies between 20 to 40 for each unloading bay. For each scenario the experiment will continue as long as all the orders have been served, and after each run of the task allocation algorithm the warehouse status is updated removing the transportation tasks that have been executed and updating the demand for those that have been partially completed (see Section 3.2). The new warehouse status is the new input for the task allocation algorithm (i.e., BMS or DSA). Hence for each scenario we have several runs of the task assignment algorithms.

Figure 6 reports an example of the warehouse topology scheme used in our experiments while Figure 7 reports a screenshot from the Gazebo simulator for a specific example with 5 loading bays, 3 unloading bays and 2 robots. The same warehouse logistic scenario is then solved by both the BMS and the DSA algorithm.

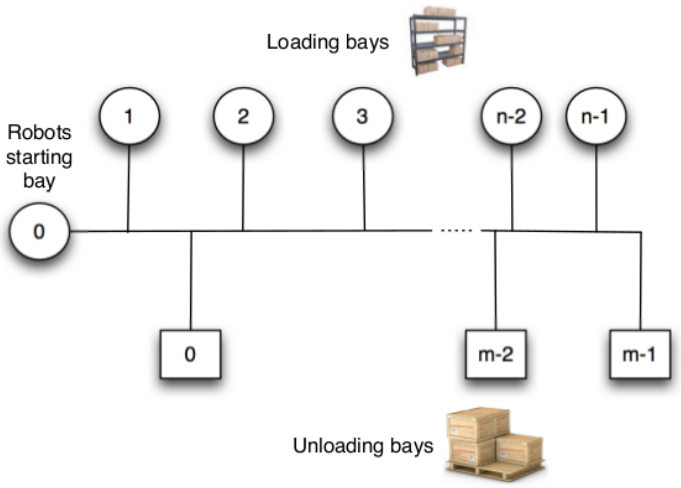

Figure 6: Warehouse topology scheme.

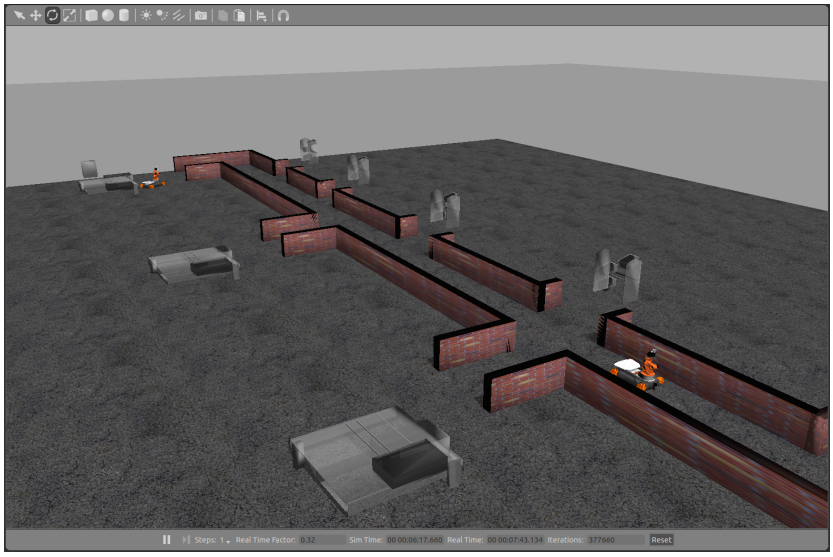

Figure 7: Warehouse topology example in Gazebo, with 5 loading bays, 3 unloading bays and 2 robots.

The two algorithms has been compared using scenarios similar to those described in Section 2.1 where the number of loading bays is higher than the number of unloading bays. This is a typical situation for warehouses, as usually

a wholesaler has several items types (e.g., books) but it will typically have few carriers that distribute the items in the local area. To evaluate our algorithm in different operative conditions and to assess how the various parameters influence the performance of the system, we vary the number of loading/unloading bays and the number of robots in the system. In more detail, we used seven scenarios described in Table 1.

	# loading bays	**# unloading bays**	**# robots**
Scenario 1	5	3	2
Scenario 2	20	7	5
Scenario 3	20	5	5
Scenario 4	30	20	20
Scenario 5	30	20	15
Scenario 6	30	20	10
Scenario 7	20	5	10

Table 1: Main parameters for the scenarios used in the experiments.

To assess the performance of the approaches we consider the following measures:

1. The average robot interferences: the average number of robots using the same line.
2. The effective interactions: the highest number of robot using the same line.
3. The Average Task Completion Time (ATCT): the time required to complete a transportation task.

The first two metrics describe the interferences on the routes and are connected with task throughput. That is, the higher the interferences the higher is the travel time for moving from a bay to another. The third metric is a direct measure of the throughput (the lower the better).

4.2. Results

Tables 2, 3, 4 report the above defined metrics for each scenario. Our results show that BMS is more effective than DSA for what concerns robot interactions. Such better performance requires a higher number of messages exchanged, nonetheless, these numbers are not problematic for the bandwidth of communication networks typically used in robotics (e.g., IEEE 802.11). The BMS usually does not improve massively the number of robots over each line on average (average interferences), but it fully shows its potential in limiting the maximum number of interferences and reducing the average completion time for the tasks. In other words, the BMS avoids critical situation for robot interferences where several robots attempt to use the same route. Comparing the Table 3 and 4 we can see that, when the number of robots and loading bays are fixed, the performance improvement of BMS over DSA significantly increases with the number

of unloading bays. This is because BMS is able to find assignments of higher quality that better distribute robots over the available routes, hence reducing robot interferences.

	Avg. robot int.	**Effective interactions**	**ATCT (s)**
BMS	1,098	1,353	33,72727
DSA	1,299	1,750	45,21429
BMS imp.	18,30%	29,32%	34,06%

Table 2: Results of the scenario with 2 robots, 3 unloading bays, 5 loading bays. Comparison between the DSA and BMS performances.

	Avg. robot int.	**Effective interactions**	**ATCT (s)**
BMS	1,332	2,130	40,30989
DSA	1,836	3,429	62,20257
BMS imp.	37,81%	60,93%	54,31%

Table 3: Results of the scenario with 5 robots, 20 loading bays, 7 unloading bays. Comparison between the DSA and BMS performances.

	Avg. robot int.	**Effective interactions**	**ATCT (s)**
BMS	1,635	2,565	40,30989
DSA	2,100	3,842	50,67556
BMS imp.	28,49%	49,78%	25,71%

Table 4: Results of the scenario with 5 robots, 20 loading bays, 5 unloading bays. Comparison between the DSA and BMS performances.

Experiments on scenarios 4, 5, 6 are designed to investigate both the run time for the two algorithms and the time needed for completing a single transportation task. Specifically, Table 5 reports results for the run time. Results show that the DSA is significantly faster than BMS. However BMS can still solve the problem in seconds hence being suitable for on-line task assignment. In fact, the computation time is significantly smaller than the time required by the robots to complete a task (see Table 5).

Figure 8 compares the average time to complete a transportation task for DSA and BMS. In this experiment the maximum speed for robots is 0.5 m/s and the average distance between loading and unloading bays is about 3 meters. Results show that DSA always incurs a higher completion time. This is because it finds solutions of lower quality with respect to BMS, hence the chance that

a robot must to stop to avoid collisions with other robots is higher[5]. Moreover, notice that the improvement in performance between BMS and DSA increases when the number of robots increases (e.g., with 20 robots the completion time for BMS is almost half of the completion time for DSA). This confirms that BMS can better manage the interactions between robots significantly improving the throughput of the system.

Robots	**DSA Run time (s)**	**BMS Run time (s)**
10	0,01	1,98
15	0,02	2,13
20	0,03	2,47

Table 5: Run time for the task allocation algorithms.

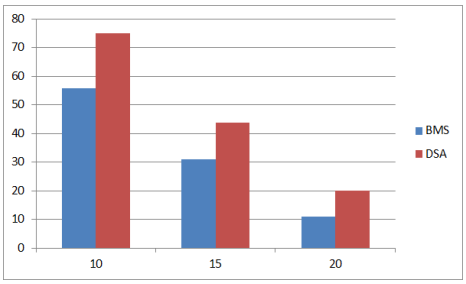

Figure 8: Average completion time for a transportation task, the y axis reports time in seconds, the x axis reports the number of robots.

Finally, in scenario 7 we have more robots than unloading bays, hence the possibility of having interferences between robots is higher, as more robots might be allocated to the same unloading bay. Table 6 shows that in this situation the improvement in performance for BMS is more evident in terms of average robot interference and effective interactions.

5. Conclusions

This paper investigates the use of modern coordination approaches, based on graphical models, for task assignment in MRS. Specifically, we consider the task assignment problem related to warehouse logistic operations, where robots must transport materials from loading bays to unloading bays while minimizing

[5]Notice that, when two or more robots could collide, the simulation provides a random order of priority where just one robot per time can move until all the possible collisions are resolved.

	Avg. robot int.	Effective interactions	ATCT (s)
BMS	3,2	4,0419	14,5295
DSA	5,2215	5,75	16,03553
BMS imp.	63,17%	42,26%	10,37%

Table 6: Results for the scenario with 10 robots, 20 loading bays, 5 unloading bays. Comparison between the DSA and BMS performances.

interferences. In more detail, we provide a broad analysis of previous approaches for MRS coordination, focusing on task assignment, and we present a detailed description of graphical model approaches to coordination. Crucially, we devise a specific DCOP model for our task assignment problem, where variable are all binary and constraints are only THOPs. This allows us to use the BMS approach to efficiently solve our task assignment problem. Finally, we evaluate our approach in a simulation environment, comparing the BMS approach to a greedy local solution (i.e., DSA). Results show that BMS provides superior performances.

We believe that our work takes a first important step towards the use of graphical models for MRS task assignment in logistic scenarios, opening up a novel promising direction for MRS coordination in industrial domains.

References

[1] Bekris, K., Tsianos, K., Kavraki, L., 2009. Safe and distributed kinodynamic replanning for vehicular networks. Mobile Networks and Applications 14 (3), 292–308.

[2] Berhault, M., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P., Kleywegt, A., 2003. Robot exploration with combinatorial auctions. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Vol. 2. pp. 1957–1962.

[3] Bishop, C. M., 2006. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[4] Boscolo, N., Battisti, R., Munaro, M., Farinelli, A., Pagello, E., 2013. A distributed kinodynamic collision avoidance system under ROS. In: Lee, S., Cho, H., Yoon, K.-J., Lee, J. (Eds.), Intelligent Autonomous Systems 12. Vol. 194 of Advances in Intelligent Systems and Computing. Springer Berlin Heidelberg, pp. 511–521.

[5] Cao, Y. U., Fukunaga, A., Kahng, A., 1997. Cooperative mobile robotics: Antecedents and directions. Autonomous Robots 4, 1–23.

[6] Cerquides, J., Farinelli, A., Meseguer, P., Ramchurn, S. D., 2014. A tutorial on optimization for multi-agent systems. The Computer Journal 57 (6), 799–824.

[7] Dechter, R., 2003. Constraint Processing. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[8] Dias, M., Zlot, R., Kalra, N., Stentz, A., July 2006. Market-based multirobot coordination: A survey and analysis. Proceedings of the IEEE 94 (7), 1257–1270.

[9] Farinelli, A., Iocchi, L., Nardi, D., 2004. Multirobot systems: A classification focused on coordination. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 34 (5), 2015–2028.

[10] Farinelli, A., Iocchi, L., Nardi, D., Ziparo, V. A., 2006. Assignment of dynamically perceived tasks by token passing in multi-robot systems. Proceedings of the IEEE, Special issue on Multi-Robot Systems 94 (7), 1271–1288, iSSN:0018-9219.

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

[12] Farinelli, A., Rogers, A., Petcu, A., Jennings, N. R., 2008. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In: Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008). pp. 639–646.

[13] Fave, F. M. D., Rogers, A., Xu, Z., Sukkarieh, S., Jennings, N., May 2012. Deploying the max-sum algorithm for coordination and task allocation of unmanned aerial vehicles for live aerial imagery collection. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp. 469–476.

[14] Ferreira, Jr., P. R., Dos Santos, F., Bazzan, A. L., Epstein, D., Waskow, S. J., May 2010. Robocup rescue as multiagent task allocation among teams: Experiments with task interdependencies. Autonomous Agents and Multi-Agent Systems 20 (3), 421–443.

[15] Fitzpatrick, S., Meertens, L., 2003. Distributed Sensor Networks A multiagent perspective. Kluwer Academic, Ch. Distributed Coordination through Anarchic Optimization, pp. 257–293.

[16] Frey, B. J., Dueck, D., 2007. Clustering by passing messages between data points. Science 315 (5814), 972–976.

[17] Gerkey, B. P., Matarić, M. J., 2004. A formal analysis and taxonomy of task allocation in multi-robot systems. International Journal of Robotics Research 23 (9), 939–954.

[18] Kask, K., Dechter, R., Larrosa, J., Dechter, A., 2005. Unifying tree decompositions for reasoning in graphical models. Artificial Intelligence 166 (12), 165 – 193.

[19] Koenig, C., Tovey, M., Lagoudakis, V., Markakis, D., Kempe, P., Keskinocak, Kleywegt, A., Meyerson, A., Jain, S., 2006. The power of sequential single-item auctions for agent coordination [nectar paper]. In: In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). pp. 1625–1629.

[20] Koes, M., Nourbakhsh, I., Sycara, K., June 2005. Heterogeneous multirobot coordination with spatial and temporal constraints. In: Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI). AAAI Press, pp. 1292–1297.

[21] Kok, J. R., Vlassis, N., July 2005. Using the max-plus algorithm for multiagent decision making in coordination graphs. In: RoboCup-2005: Robot Soccer World Cup IX. Osaka, Japan.

[22] Korsah, G., Stentz, A., Dias, M., 2013. A comprehensive taxonomy for multi-robot task allocation. International Journal of Robotics Research 32 (12), 1495–1512.

[23] Kschischang, F. R., Frey, B. J., Loeliger, H. A., 2001. Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 42 (2), 498–519.

[24] Kube, C. R., Bonabeau, E., 2000. Cooperative transport by ants and robots. Robotics and Autonomous Systems 30 (1), 85–101.

[25] Modi, P. J., Shen, W.-M., Tambe, M., Yokoo, M., 2005. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. Artificial Intelligence 161 (12), 149 – 180, distributed Constraint Satisfaction.

[26] Nanjanath, M., Gini, M., Jul. 2010. Repeated auctions for robust task execution by a robot team. Robot. Auton. Syst. 58 (7), 900–909.

[27] Parker, L. E., April 1998. ALLIANCE: An architecture for fault tolerant multirobot cooperation. IEEE Transactions on Robotics and Automation 14 (2), 220–240.

[28] Petcu, A., Faltings, B., 2005. DPOP: A scalable method for multiagent constraint optimization. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, (IJCAI 2005). pp. 266–271.

[29] Pippin, C., Christensen, H., Weiss, L., 2013. Performance based task assignment in multi-robot patrolling. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing. SAC '13. ACM, New York, NY, USA, pp. 70–76.

[30] Pujol-Gonzalez, M., Cerquides, J., Farinelli, A., Meseguer, P., Rodriguez-Aguilar, J. A., 2015. Efficient Inter-Team Task Allocation in RoboCup Rescue. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. AAMAS '15. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 413–421.

[31] Pujol-Gonzalez, M., Cerquides, J., Meseguer, P., Rodrguez-Aguilar, J., Tambe, M., 2013. Engineering the decentralized coordination of UAVs with limited communication range. In: Bielza, C., Salmern, A., Alonso-Betanzos, A., Hidalgo, J., Martnez, L., Troncoso, A., Corchado, E., Corchado, J. (Eds.), Advances in Artificial Intelligence. Vol. 8109 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 199–208.

[32] Scerri, P., Farinelli, A., Okamoto, S., Tambe, M., 2005. Allocating tasks in extreme teams. In: Proc. of AAMAS 05. Utrecht, Netherland, pp. 727–734.

[33] Scerri, P., Xu, Y., Liao, E., Lai, G., Sycara, K., July 2004. Scaling teamwork to very large teams. In: Proceedings of AAMAS. pp. 888–895.

[34] Simmons, R., Apfelbaum, D., Burgard, W., Fox, D., Moors, M., Thrun, S., Younes, S., 2000. Coordination for multi-robot exploration and mapping. In: In proceedings of the AAAI national conference on artificial intelligence. AAAI, pp. 852–858.

[35] Tarlow, D., Givoni, I. E., Zemel, R. S., 2010. Hop-map: Efficient message passing with high order potentials. In: Proceedings of 13th Conference on Artificial Intelligence and Statistics.

[36] Tovey, C., Lagoudakis, M., Jain, S., Koenig, S., 2005. The generation of bidding rules for auction-based robot coordination. In: Parker, L., Schneider, F., Schultz, A. (Eds.), Multi-Robot Systems. From Swarms to Intelligent Automata Volume III. Springer Netherlands, pp. 3–14.

[37] Veloso, M., Stone, P., 2002. A Survey of Multiagent and Multirobot Systems. AK Peters, Ch. in Robot Teams: From Diversity to Polymorphism, T. Balch and L. E. Parker, eds.

[38] Weiss, Y., Freeman, W. T., 2001. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. IEEE Transactions on Information Theory 47 (2), 723–735.

[39] Wurman, P. R., D'Andrea, R., Mountz, M., 2007. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada. AAAI Press, pp. 1752–1760.

[40] Yu, B., Scerri, P., Sycara, K., Xu, Y., Lewis, M., 2006. Scalable and reliable data delivery in mobile ad hoc sensor networks. In: Proc. of the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).

[41] Zhang, W., Wang, G., Xing, Z., Wittenburg, L., Jan. 2005. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. Artificial Intelligence 161 (1-2), 55–87.

[42] Zlot, R., Stentz, A., Dias, M., Thayer, S., 2002. Multi-robot exploration controlled by a market economy. In: Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on. Vol. 3. pp. 3016–3023.