Optimal Temporal Decoupling in Multiagent Systems

Léon R. Planken
Delft University of Technology
Delft, The Netherlands
L.R.Planken@tudelft.nl

Mathijs M. de Weerdt
Delft University of Technology
Delft, The Netherlands
M.M.deWeerdt@tudelft.nl

Cees Witteveen
Delft University of Technology
Delft, The Netherlands
C.Witteveen@tudelft.nl

ABSTRACT

When agents need to interact in order to solve some (possibly common) problem, resolving potential conflicts beforehand is often preferred to coordination during execution. Agents may lose some flexibility, but their course of action will be more predictable and often also more efficient, obtaining a socially optimal outcome instead of a local optimum. One way to resolve conflicts beforehand is to give extra constraints to each of the agents such that when they all meet these constraints, the resulting execution is conflict-free. A set of constraints that meets this requirement is called a *decoupling* of the original problem; if it also maximizes the social welfare (i.e. the sum of the valuations of all the agents), it is called optimal. Representing interesting multiagent problems as a constraint problem, we show that finding an optimal decoupling is at least as hard as finding a solution for the constraint problem. We therefore focus on a constraint problem that is efficiently solvable, but still very relevant and interesting in the context of multiple agents executing their actions, i.e. the Simple Temporal Problem (STP). Two more technical results, then, are that we resolve the open question whether finding an optimal decoupling of the STP is NP-hard (it is), and if all agents have linear valuation functions, this decoupling problem can be solved efficiently.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems; F.2 [**Analysis of Algorithms and Problem Complexity**]

General Terms

Algorithms, Theory

Keywords

Optimal Decoupling, Temporal Decoupling Problem

1. INTRODUCTION

In multiagent systems, a number of autonomous agents perform activities to solve some (possibly common) problem. There are usually many interdependencies between these activities, for example caused by shared goals, scarce resources, or consumer–producer relations. Consequently, these activities have to be *coordinated* in order to ensure a correct solution. In principle, such coordination can be achieved by solving all agents' problems together as one multiagent problem, completely coordinating all their activities in the process. We feel that this is counter to the idea of autonomy and privacy of the agents, thereby removing all flexibility, and is thus a useful approach only in rare cases. The other extreme is to coordinate during execution. For example, semaphores (e.g. traffic lights) can be used to prevent agents from using the same resource at the same time; alternatively, agents can wait for other agents to complete activities they depend upon. Such real-time coordination allows for last-moment changes and thereby offers the agents a significant amount of flexibility. Often, however, the process is more efficient (obtaining a social optimum instead of a local optimum) and results are more predictable when (most of the) coordination is performed before execution; moreover, communication during execution may be prohibitively expensive or simply impossible.

In this paper, we therefore propose to model the problem of coordinating the activities of agents beforehand by finding a *decoupling* [12, 13]. The idea behind such a decoupling is rather simple: additional constraints ensure that individual parts can be solved independently, and all combinations of (local) solutions to these parts can be merged to constitute a (global) solution of the original multiagent problem.

Example 1. Suppose that Alice and Bob plan a joint meeting. Alice is able to attend the meeting from 14:00 till 16:00 and she will stay for 45 minutes. Bob has time from 13:00 till 16:00, but he will stay for only 30 minutes. They need at least 10 minutes to talk to each other. Clearly, if Alice plans to attend the meeting from 15:15 till 16:00 and Bob chooses to attend from 13:00 till 13:30, they will not have the possibility to talk to each other. If, however, we enforce both parties to choose an arrival time between 15.00 and 16.00, they will have enough opportunity to meet. This constraint effectively decouples the overall problem into two separate subproblems in such a way that whatever schedule Alice chooses and whatever schedule Bob chooses, if they meet their own constraints, the meeting objective is satisfied.

Since a decoupling introduces new constraints, the set of solutions of the original system might be reduced. One commonly used objective is then to choose among alterna-

Cite as: Optimal Temporal Decoupling in Multiagent Systems, Planken, De Weerdt, and Witteveen, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 789-796

tive decouplings the one that minimizes the loss of flexibility (to choose alternative local solutions) introduced by the additional constraints. For example, Fitoussi and Tennenholtz [9] study how to obtain minimal social laws, which define a set of forbidden strategies such that no strategy can be removed without losing usefulness (i.e. any combination of the allowed strategies is a global solution). In this paper we generalize this notion of (local) optimality by optimizing the valuation of the agents for the chosen decoupling. Such a valuation can express other preferences than those on flexibility as well. For example, Bob may prefer an early time and Alice a later time. We then study the problem of finding a decoupling that is as good as possible for all agents. To be precise, we optimize the sum of the valuations of the agents for the chosen decoupling, i.e. the utilitarian social welfare.

In this paper, we use general constraint systems to represent many interesting multiagent problems, be they distributed constraint satisfaction, task/resource allocation, or scheduling problems. We show that finding an arbitrary decoupling is as hard as finding a solution for a constraint system. This implies that finding a decoupling in general is an NP-hard problem. Therefore, finding an optimal decoupling must be at least as hard, making it very unlikely to have an algorithm that always finds an optimal solution efficiently. We then concentrate on constraint systems where optimal solutions can be found in polynomial time. In addition, in the context of self-interested agents optimal solutions are important for a second reason; if optimality cannot be guaranteed, agents may want to lie about their valuations, thus changing the objective [17]. Therefore, concentrating on constraint systems where finding a solution is tractable, we study the Simple Temporal Problem (STP). STPs occur as a subproblem of many interesting multiagent problems and some work has already been done on decoupling of STP by Hunsberger [13]. However, whether an optimal decoupling of STP can be found efficiently was still an open problem. We show that for general valuation functions finding an optimal decoupling is NP-hard, while for linear functions this can be done in polynomial time.

2. BACKGROUND

A constraint system is a general representation for a large class of combinatorial real-life problems. Constraint systems have been used to represent and solve planning and scheduling problems, resource allocation problems and design and configuration problems [3]. The basic ingredients of a constraint system $\mathcal{S}$ are a set C of constraints over a set X of variables x_i each taking values in some finite domain D_i. A constraint system $\mathcal{S}$ is solved if we find values for each of the variables such that all the constraints are satisfied. If constraint systems are used to represent and solve problems in multiagent systems, each agent A_i has its own disjoint set $X_i \subseteq X$ of variables to assign values to. The (common) task of the agents is then to find suitable values to the variables such that all constraints are satisfied.

We would like to respect the agents' privacy, autonomy, and flexibility by allowing them to solve their subproblem completely independently from the others. We are therefore interested in *decoupling* techniques for constraint systems. Such a decoupling ensures that the local solutions determined by each of the agents can always be *merged* to yield a solution to the complete system.

Although in constraint systems decomposition is a common technique to split a problem in a number of parts in such a way that the global solution can be efficiently assembled from the solutions of the parts [10, 16, 22], this approach differs from the decoupling approach mentioned above. Firstly, in the decomposition approach the structure of the problem (i.e. the set of constraints) dictates the way in which the subproblems are generated (for example by guaranteeing that the subproblems are acyclic [5]), while in the decoupling approach the subproblems are dictated by the given partitioning of the variables due to the agents' span of control. Secondly, in the decomposition approach, the subproblems generated are usually not independently solvable, because the subproblems do not need to be completely disjoint. Before we state this decoupling problem in a more precise way, we first introduce some notational conventions.

3. PRELIMINARIES

In this section we formally define constraint systems, distributed constraint systems, and decoupling of distributed constraint systems. Then, in the next section, we show that finding a decoupling is as hard as solving a constraint problem.

A constraint system $\mathcal{S} = \langle X, D, C\rangle$ where X is a (finite) set of variables, D is a set of (value) domains D^i for every variable $x_i \in X$ and C is a set of constraints on X. We assume constraints $c \in C$ to be specified as formulas over some language.[1] A solution s of the system is an assignment $s = \{x_i \leftarrow d_i\}_{i=1}^n$ to all variables in X such that each $c \in C$ is satisfied. The set of such solutions s is denoted by $Sol(\mathcal{S})$.

The system $\mathcal{S}$ is called *consistent* if $Sol(\mathcal{S}) \neq \emptyset$. We assume the set of solutions $Sol(\mathcal{S})$ to be anti-monotonic in the set of constraints; that is, if $\mathcal{S} = \langle X, D, C\rangle$ and $\mathcal{S}' = \langle X, D, C'\rangle$ are such that $C \subseteq C'$, then $Sol(\mathcal{S}') \subseteq Sol(\mathcal{S})$. For every $c \in C$, let $Var(c)$ denote the set of variables mentioned in c. Given a set of constraints C and a set of variables X', we let $C_{X'}$ denote the subset $\{c \in C \mid Var(c) \subseteq X'\}$.

In this paper we consider constraint systems $\mathcal{S}$ that are *distributed* [24]; that is, there is a set of agents A_i, each being able to make assignments or add relations to a *subset* X_i of the variables. More specifically, the collection $\{X_i\}_{i=1}^N$ constitutes a partitioning of X, i.e. $\bigcup_{i=1}^N X_i = X$, while for $1 \leq i < j \leq N$, $X_i \cap X_j = \emptyset$. Given such a partitioning $\{X_i\}_{i=1}^N$, $\mathcal{S}_i = \langle X_i, D_{X_i}, C_{X_i}\rangle$ is the subsystem that has to be solved by agent A_i, where D_{X_i} is the set of domains for the variables in X_i, and C_{X_i} is as defined above.

We are interested in those distributed constraint systems where each agent A_i is able to find an (arbitrary) solution s_i for the subsystem $\mathcal{S}_i = \langle X_i, D_{X_i}, C_{X_i}\rangle$ controlled by A_i in such a way that the simple merging $s = s_1 \sqcup s_2 \sqcup \ldots \sqcup s_N$ of these individual solutions always is a solution of the original system $\mathcal{S}$. If a distributed constraint system $\mathcal{S} = \langle\{X_i\}_{i=1}^N, D, C\rangle$ satisfies this property, we say that $\mathcal{S}$ is a *decoupled* constraint system.

Usually, distributed constraint systems do not satisfy this decoupling property. However, a decoupled version can be obtained by adding extra constraints. In particular, a distributed constraint system $\mathcal{S}' = \langle\{X_i\}_{i=1}^N, D, C'\rangle$ is a *decoupling* of a distributed constraint system $\mathcal{S} = \langle\{X_i\}_{i=1}^N, D, C\rangle$ if (i) $C \subseteq C'$ and (ii) $\mathcal{S}' = \langle\{X_i\}_{i=1}^N, D, C'\rangle$ is a decou-

[1]To preserve generality, we don't feel the need to specify the set of allowable operators used in the constraints $c \in C$ and their interpretation.

pled constraint system. Notice that, by anti-monotonicity of $Sol(\mathcal{S})$ in C, it follows that every solution of $\mathcal{S}'$ is also a solution of $\mathcal{S}$.

4. DECOUPLING AND SOLVING A CONSTRAINT SYSTEM

One of the basic problems in decoupling is how to find a decoupling of a given distributed constraint system. It has been argued that for general constraint systems, finding a decoupled version of a distributed constraint system is intractable [23]: more in particular, that finding a decoupling for a distributed variant of a constraint system $\mathcal{S}$ is as hard as finding a solution to $\mathcal{S}$. Here, we present a more detailed and complete version of this result.

PROPOSITION 1. *Let $\mathcal{C}$ be an arbitrary class of constraint systems allowing at least equality constraints. Then there exists a polynomial algorithm to find a solution for constraint systems in $\mathcal{C}$ if and only if there exists a polynomial algorithm that, given a constraint system $\mathcal{S} \in \mathcal{C}$ and an arbitrary partition $\{X_i\}_{i=1}^N$ of X, finds a decoupling of $\mathcal{S} = \langle \{X_i\}_{i=1}^N, D, C\rangle$.*

PROOF. Suppose that there exists a polynomial algorithm A to find a solution for constraint systems in $\mathcal{C}$. We show how to construct a polynomial algorithm for finding a decoupling for an arbitrary partition of such a constraint system. Let $\mathcal{S} \in \mathcal{C}$ be a constraint system and $\{X_i\}_{i=1}^N$ an arbitrary partitioning of X. To obtain a decoupling $\mathcal{S}'' = (\{X_i\}_{i=1}^N, D, C')$ of $\mathcal{S}' = (\{X_i\}_{i=1}^N, D, C)$, first, using A, we compute a solution s of $\mathcal{S}$. For every X_i, let $C_{s_i} = \{x = d \mid x \leftarrow d \in s, x \in X_i\}$ be a set of unary constraints for variables in X_i directly obtained from s. Then the decoupled subsystems (X_i, D_i, C_i') of $\mathcal{S}''$ are simply obtained by setting $D_i = D_{X_i}$ and $C_i' = C_{X_i} \cup C_{s_i}$. Note that each of these subsystems (X_i, D_i, C_i') has a unique solution $s_i = \{x \leftarrow d \in s \mid x \in X_i\}$ and the merging of these solutions s_i equals s, i.e. a solution to the original system $\mathcal{S}$. Therefore $\mathcal{S}''$ is a decoupling for $\mathcal{S}$ that can be obtained in polynomial time.

Conversely, suppose we can find a decoupling $\mathcal{S}''= (\{X_i\}_{i=1}^N, D, C')$ for any distributed version $\mathcal{S}' = (\{X_i\}_{i=1}^N, D, C)$ of a constraint system $\mathcal{S} = \langle X, D, C\rangle \in C$ in polynomial time. We show how to obtain a solution s of $\mathcal{S}$ in polynomial time. Since the decoupling $\mathcal{S}''$ can be obtained for any partitioning of X, we choose the partitioning $\{X_i\}_{i=1}^N$ where $X_i = \{x_i\}$ for $i = 1, 2, \ldots, N$. Since the decoupling $\mathcal{S}'' = (\{X_i\}_{i=1}^N, D, C')$ has been obtained in polynomial time, it follows that $|C'|$ is polynomially bounded in the size of the input $\mathcal{S}'$. Hence, the resulting decoupled subsystems $\mathcal{S}''_{\{x_i\}}$ of $\mathcal{S}''$ each consist of a polynomially bounded set of *unary* constraints. It is well known that such constraint systems are solvable in polynomial time [2]. Therefore, in polynomial time for each subsystem $\mathcal{S}''_{\{x_i\}}$ an arbitrary value $d_i \in D_{\{x_i\}}$ for x_i can be obtained, satisfying all constraints. Let $s_i = \{x_i \leftarrow d_i\}$ denote the solution obtained for $\mathcal{S}''_{\{x_i\}}$. Since $\mathcal{S}''$ is a decoupled system, the merging $s = s_1 \sqcup s_2 \sqcup \ldots \sqcup s_N$ must be a solution of $\mathcal{S}''$ as well. Therefore, s is a solution of $\mathcal{S}$, too. Hence, given a polynomial algorithm for decoupling, we can compose a solution $s \in Sol(\mathcal{S})$ in polynomial time. □

It is well-known that for general constraint systems, finding a solution is NP-hard [3]. Therefore, finding a decomposition, even a trivial one, is exactly as hard. Therefore, whatever notion of *optimality* we would like to introduce for decoupling, finding an optimal decoupling must be as hard. In [23] it has been shown that using various notions of optimality, finding an optimal decoupling is sometimes even essentially harder (under the usual complexity assumptions) than finding an arbitrary one.

Since we are especially interested in finding (optimal) decouplings in an efficient way, the results stated above force us to look at subclasses of constraint systems that can be solved efficiently. We therefore concentrate on an interesting tractable class of constraint systems: the Simple Temporal Problem.

5. SIMPLE TEMPORAL PROBLEM

The Simple Temporal Problem (STP) is about finding out whether there exists a solution for a special type of constraint system $\mathcal{S} = \langle X, D, C\rangle$. Here, each $x_i \in X$ is a temporal variable, taking values in some temporal domain (usually the set of real numbers), and C consists of linear constraints of the form $c_{i\to j} : x_j - x_i \leq w_{i\to j}$, where $w_{i\to j}$ is some real constant. Since the domains for all the variables are the same (a fixed temporal domain), we usually assume D to be known and define an instance of the STP $\mathcal{S}$ as a tuple $\langle X, C\rangle$. Unary constraints $x_i \in [a_i, b_i]$ with $a_i, b_i \in D$ are usually represented in the STP with the help of an additional special time point z, called the temporal reference point. This variable indicates some fixed point in time and always takes the value 0 in a solution. Each unary constraint then takes the form of a pair of constraints $c_{x_i\to z}$ and $c_{z\to x_i}$ where $w_{x_i\to z} = -a_i$ and $w_{z\to x_i} = b_i$.

Henceforth, we also use the term Simple Temporal Network (STN) to refer to instances of the STP. The problem of finding out whether there exists a solution to an STN, and if so, to find a solution for such an STN can be solved efficiently [4]. The solution to an STN can be viewed as a *schedule* for some set of constrained activities as specified in the STN.

To introduce a distributed version of an STP, assume that we have agents 1 through N, who want to solve (or execute) a Simple Temporal Problem (STP) $\mathcal{S} = \langle X, C\rangle$ together. The time-point variables $X \setminus \{z\}$ are partitioned into N subsets $X_1, \ldots, X_N$, such that for $i \neq j : X_i \cap X_j = \{z\}$; that is, the intersection of each pair of subsets always contains just the special time-point variable z. Such a partition is called a z-partition. Every agent i is given control over all time-point variables in X_i with the exception of z, which is always taken to be 0. Given such a distributed STP, the *Temporal Decoupling* (TD) problem, informally speaking, is to add new constraints C' to $\mathcal{S}$, guaranteeing that each agent i can execute the part of the STP instance restricted to its subset of time-point variables X_i, i.e. $\mathcal{S}_i = \langle X_i, (C \cup C')_{X_i}\rangle$ independently of the other agents without the risk of running into inconsistency. Note that Proposition 1 is applicable[2] to the STP; thus, the TD problem, like the STP itself, is tractable.

Hunsberger [13] was the first to study Temporal Decoupling. By introducing the notion of *flexibility* in STPs he was able to define an optimization variant of the TD problem: an optimal TD is a decoupling of a constraint system preserv-

[2]An (implicit) equality constraint $x_i = c_i$ can be obtained by combining two unary constraints with weights $w_{z\to x_i} = c_i$ and $w_{x_i\to z} = -c_i$.

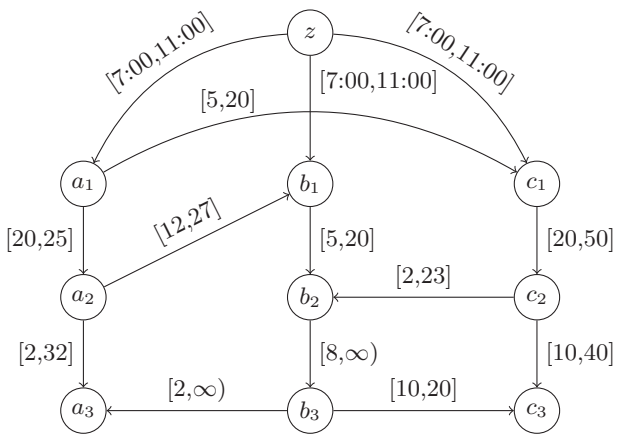

Figure 1: The three chemical processes, depicted in an STN

ing maximal flexibility. Roughly, Hunsberger considered a decoupling $\mathcal{S}'$ as a *maximal flexibility* preserving decoupling of a distributed constraint system $\mathcal{S}$ if there does not exist an alternative decoupling $\mathcal{S}''$ for $\mathcal{S}$ where *(i)* all constraints of $\mathcal{S}''$ are at least as tight as the constraints in $\mathcal{S}'$ and *(ii)* at least one constraint is less tight than the corresponding constraint in $\mathcal{S}'$. Hunsberger was able to find a polynomial algorithm for obtaining such a (locally) maximal flexibility preserving decoupling.

Instead of flexibility, one could easily choose other objectives for a decoupling to be optimized. Therefore, Hunsberger defined a general notion of the optimization variant of the decoupling problem by using a general metric h as the specification of the objective function to be optimized. Here, the only requirement for h is to be a real-valued function that takes an STN $\mathcal{S}$ and a decoupling $\mathcal{S}'$ for $\mathcal{S}$ as input. Hunsberger has not addressed the complexity of finding globally optimal decouplings using such a given metric h. In the next section, we show that for general metrics h this problem is NP-hard. Then, we address the problem of specifying special metrics h such that the optimal decoupling problem can be solved efficiently. We conclude this section with an elaborate example of both STP and Temporal Decoupling. In later sections, we build upon this example to demonstrate our results on Optimal Temporal Decoupling and to illustrate the use of mechanism design in the context of rational agents with private information.

Example 2. Three processes in the chemical industry are controlled by separate agents A, B, and C. Apart from the main marketable product, each of these processes produces some by-products that are of no immediate use to the agent and have little market value. However, it so happens that these three processes mutually benefit from each other's by-products. The agents have decided to work in synergy and run their processes in tight cooperation. Without this synergy, each by-product would have to be disposed of by its producing agent as little more than waste. The consuming agent, for his part, would have to buy it at some cost dominated by secondary components like shipping and handling, and therefore much higher than its intrinsic low value.

The process is pictorially represented in Figure 2. In this figure, each of the vertices labelled a_i indicates the end of

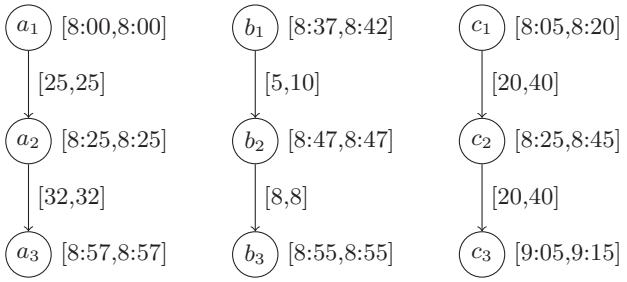

Figure 2: A possible decoupling of the example STN

a phase in the process of agent A (similarly for B and C). Furthermore, each of the industries starts at some time between 7:00 and 11:00 AM, denoted by the arc from z; this *temporal reference point* is taken to stand for the beginning of the day, at midnight. The arc between a_1 and a_2 labelled with an interval of $[20, 25]$ indicates that the end of phase a_2 must occur at least 20 minutes and at most 25 minutes after the end a_1; the remaining arcs can be interpreted likewise.

It is of utmost importance that this global plan be adhered to. Violation of any of the constraints may mean that substances can no longer be used because they have expired, resulting certainly in large extra costs, but possibly also in damage to equipment. To ensure global consistency, a possibility would be for each agent to relinquish control to a central authority which then runs all three of the processes. However, each agent would rather retain control of its own process. As we have seen before, this means that we have to find a decoupling: a set of additional local constraints for each agent that, given compliance with the local constraints, guarantees consistency of the global problem instance. Once such a decoupling has been found, inter-agent constraints can be ignored.

A possible decoupling of our example problem is given in Figure 2. To avoid clutter, we have omitted z and the arcs between it and the remaining vertices in this representation; the labels of the omitted arcs are instead placed alongside each of the vertices. The reader can verify that this is indeed a decoupling of the original network; any solution of this network is also a solution to the original problem instance.

Note that many other decouplings exist, and agents may strongly prefer one decoupling to another. In particular, agent A is none too happy with the decoupling as presented, as we shall soon see. We model agents' preferences over decouplings with *valuations*. Agent A prefers to start early, and values each minute that a_1 takes place before 11:00 AM at \$0.10. It also desires as short a time span as possible between a_1 and a_2, because with delay, the intermediate product loses malleability; each minute earlier is worth as much as \$100, with a maximum of \$500 for the minimum delay. Agent B, in contrast, wants the time span between b_2 and b_3 to be as long as possible, because this product becomes easier to handle with a longer delay. Each minute longer than the minimum wait time of 8 minutes is valued at \$15. Finally, all agents value flexibility in their constraints. This is measured by the total size of the intervals labelling arcs between z and each of the phases of an agent's process; for simplicity, we assume that these flexibilities are uniformly valued. For agent A, flexibility is worth \$0.50 per minute; for agent B, \$2 per minute; and for agent C (who doesn't

care about start or end times), $10 per minute. It can then be verified that decoupling as presented in Figure 2 is valued at $12, $10 and $450 by agents A, B, and C, respectively.

The decoupling in the example above is just one possible decoupling, which happens to be very good only for agent C. A natural question to ask here is whether it is possible to find a socially optimal decoupling that is as good as possible for all agents. This is exactly the topic of the next two sections.

6. OPTIMAL TEMPORAL DECOUPLING

Hunsberger phrases Optimal Temporal Decoupling (OTD) as an optimization problem that maximizes some metric h. This metric takes as input the original and decoupled STNs and gives a measure of this decoupling's "preferredness". This may be any function, but a good example would be a combination of the agents' local preferences on their part of the STP, e.g. the valuations as stated in Example 2. To establish the complexity, we state the decision variant, using a threshold of at least α preferredness for the metric h. Like Hunsberger, we consider here the case $N = 2$; our result then extends to the general case.

Definition 1. Let $\mathcal{S} = \langle X, C\rangle$ be an STP instance, where X is z-partitioned in X_1 and X_2. Further, let h be a temporal decoupling metric as defined by Hunsberger and let $\alpha \in \mathbb{R}$ be a constant. Then, the OPTIMAL TEMPORAL DECOUPLING problem is to decide whether there exists a decoupling $(\mathcal{S}_1, \mathcal{S}_2)$ of $\mathcal{S}$ such that $h(\mathcal{S}, \mathcal{S}_1, \mathcal{S}_2) \geq \alpha$.

The following result solves an open problem as mentioned by Hunsberger [13]:

THEOREM 2. OPTIMAL TEMPORAL DECOUPLING *is* NP*-hard.*

In the proof of Theorem 2, we use a (linear) reduction from VERTEX COVER.

Let $I = \langle G = \langle V, E\rangle, K\rangle$ be a instance of VERTEX COVER, where $|V| = n$ and vertices are denoted by the natural numbers $\{1, \ldots, n\}$. Define the following OTD instance $R(I)$:

$$\begin{aligned} X_1 &= \{z\} \cup \{x_{2i-1} \mid i \in V\} \\ X_2 &= \{z\} \cup \{x_{2i} \mid i \in V\} \\ C &= \{x_{2i} - x_{2i-1} \in [0,1] \mid i \in V\} \\ &\cup \{z - x_{2i-1} \in [0,1] \mid i \in V\} \\ &\cup \{x_{2i} - z \in [0,1] \mid i \in V\} \end{aligned}$$

To decouple this instance, we are looking for sets of constraints C_1, C_2, ranging over the time-point variables X_1, X_2 respectively. Now, we define the metric h over a n-dimensional vector $\mathbf{w}$, where each element w_i takes a value c for $(z - x_{2i-1} \leq c) \in C_1$. Note that to induce a valid decoupling, all w_i must be in $[0, 1]$. Setting the optimality bound $\alpha = n - K$, we define the metric as follows:

$$h(\mathbf{w}) = \sum_{i \in V} w_i \; - \; n^2 \sum_{i \in V} w_i(1 - w_i) \; - \; n \sum_{\{i,j\} \in E} w_i w_j \qquad (1)$$

The idea of the reduction is that $w_i = 0$ if and only if v_i is in the vertex cover.

PROOF OF THEOREM 2. Let $V' \subseteq V$ be a vertex cover of G with $|V'| = K$. We set $C_2 = \{x_{2i} - z = 0 \mid i \in V\}$ so that each element w_i of $\mathbf{w}$ can be set to any value in $[0, 1]$ to induce a valid decoupling. For each $i \in V$, set $w_i = 0$ if $i \in V'$, $w_i = 1$ otherwise. Since V' is a vertex cover, it holds for all edges $\{i, j\} \in E$ that $i \in V' \vee j \in V'$, and thus $w_i = 0 \vee w_j = 0$. Then, the following holds for the summations constituting h:

$$\begin{aligned} \sum_{i \in V} w_i &= n - K \\ \sum_{i \in V} w_i(1 - w_i) &= 0 \\ \sum_{\{i,j\} \in E} w_i w_j &= 0 \end{aligned}$$

So $h(w_1, \ldots, w_n) = n - K = \alpha$.

Before showing the converse, let us first examine the metric h a little closer, w.l.o.g. taking the partial derivative $\partial_{w_1} h$:

$$\partial_{w_1} h(\mathbf{w}) = 1 - n^2(1 - 2w_1) - n \sum_{\{1,j\} \in E} w_j$$

For $n > 1$, it can be seen that $\partial_{w_1} h(0, \ldots, w_n) < 0$; also, noting that $\sum_{\{1,j\} \in E} w_j < n$, it follows that $\partial_{w_1} h(1, \ldots, w_n) > 0$. Since this argument can be repeated for all dimensions of $\mathbf{w}$, and h is a smooth function, we conclude that it has a minimum in the interior and its value is highest along the edges where the variables take values from $\{0, 1\}$.

Now, let $\mathbf{w}$ be a vector for which $h(\mathbf{w}) \geq n - K$. By the preceding discussion, any element of $\mathbf{w}$ with $0 < w_i < 1$ can be replaced by a value $\hat{w}_i \in \{0, 1\}$ to produce a $\{0, 1\}$-valued vector $\hat{\mathbf{w}}$ with $h(\hat{\mathbf{w}}) > h(\mathbf{w})$. This implies that the second term of Equation 1 is zero. The last term of Equation 1 must also be zero; otherwise, $h(\hat{\mathbf{w}}) \leq 0 < n - K$ (assuming the nontrivial case that $K < n$).

From $\hat{\mathbf{w}}$, we can now construct a set of vertices $V' = \{i \in V \mid \hat{w}_i = 0\}$. Because the last term of Equation 1 is zero, we know that V' is a vertex cover, and from $\sum_{i \in V} \hat{w}_i \geq n - K$, we have that $|V'| \leq K$. □

We have thus shown that even for only two agents and quadratic temporal decoupling metrics, solving OTD is intractable. In the next section, we show that if we limit the allowable metrics, we can efficiently solve OTD regardless of the number of agents.

7. SOLVING OTD EFFICIENTLY

Hunsberger [13] relegates the efficient solution of OTD to future research. We now show that for suitable metrics h, OTD can be solved in polynomial time, following a linear programming (LP) approach. As we shall see, the variable vector in our LP formulation represents an STP instance, and the linear inequalities it is subjected to are chosen such that the set of feasible solutions coincides with the set of all possible decouplings of some STP instance.

Let $\mathcal{S} = \langle X, C\rangle$ be the STP instance under scrutiny; furthermore, let N be the set of agents and let $\{X_i\}_{i \in N}$ be a z-partition of X, and let h be the decoupling metric. As variable vector for our LP instance, we use $\mathbf{p} \in \mathbb{R}^{X^2}$. This vector can be interpreted as an STP instance $\mathcal{S}' = \langle X, C'\rangle$ with $C' = \{c_{x \to y} : y - x \leq p_{xy} \mid (x, y) \in X^2\}$. Now, we present four sets of linear inequalities on $\mathbf{p}$.

Triangle inequality (TRI)

$$\forall (v, w, x) \in X^3 : \; p_{vx} \leq p_{vw} + p_{wx}$$

Consistency (CON)

$$\forall x \in X : \; p_{xx} = 0$$

These two sets of inequalities are based on properties of the STN as described by Dechter et al. [4, p. 72].

LEMMA 3. *If* $\mathbf{p}$ *satisfies* TRI*, then* $\mathcal{S}'$ *is a minimal STN.*

PROOF. Note that the Floyd–Warshall algorithm can be used for establishing minimality:

for $k \leftarrow 1$ **to** n **do**
$\quad \forall i, j \in \{1, \ldots, n\} : d_{ij} \leftarrow \min\{d_{ij}, d_{ik} + d_{kj}\}$

Here, the values d_{ij} are initialised to the original weights $w_{i \to j}$ and thus correspond directly to the values in $\mathbf{p}$. From TRI, it then follows easily that no updates are made by the algorithm. □

LEMMA 4. *If* $\mathbf{p}$ *additionally satisfies* CON*, then* $\mathcal{S}'$ *is consistent.*

PROOF. This corresponds exactly to the procedure of "examining the sign of the diagonal elements" as stated by Dechter et al. [4, p. 72]. □

The vector $\mathbf{p}$ can now be seen as a unique representation of all STP instances equivalent to $\mathcal{S}'$ [4, p. 67]. Conversely, for any possible minimal STN, there exists a vector $\mathbf{p}$ satisfying both TRI and CON that describes it; thus, these inequalities establish a bijection between the set of minimal STNs and the set of vectors $\mathbf{p}$.

Original constraints (ORI)

$$\forall c_{x \to y} \in C : \; p_{xy} \leq w_{x \to y}$$

Decoupling (DEC)

$$\forall c_{x \to y} \in C \text{ such that } x \in X_i \wedge y \in X_j \wedge i \neq j : \\ p_{xz} + p_{zy} \leq w_{x \to y}$$

ORI requires that all constraints in $\mathcal{S}'$ are, in the parlance of Dechter et al. [4, p. 72], tighter (in the weak sense) than those in $\mathcal{S}$. Hence, $\mathcal{S}' \subseteq \mathcal{S}$, meaning that any solution to $\mathcal{S}'$ must also be a solution to $\mathcal{S}$.

Then, all these constraints together satisfy exactly the necessary and sufficient conditions of temporal decouplings as stated by Hunsberger [13, Thms. 5.10 and 5.12]. We thus have that $\mathcal{S}'$ is a decoupling of $\mathcal{S}$.

Now, Hunsberger's temporal decoupling metric h can be used as the objective function of this linear program. We then have the following result from the discussion in this section:

THEOREM 5. *If* h *can be expressed as a linear function over* $\mathbf{p}$*, then the OTD problem can be solved in polynomial time.*

Without going into details, we wish to mention here that this theorem offers some room for extension. First, note that some additional variables and inequalities may be added to the linear program to widen the scope of considered metrics; this opens the way for criteria like "min max", among others. Furthermore, tractability is not sacrificed by allowing h to be a quadratic function expressible with a positive semidefinite matrix. This means in particular that in showing that OTD is NP-hard for general quadratic metrics (Theorem 2), we have established the bounds of tractability quite narrowly.

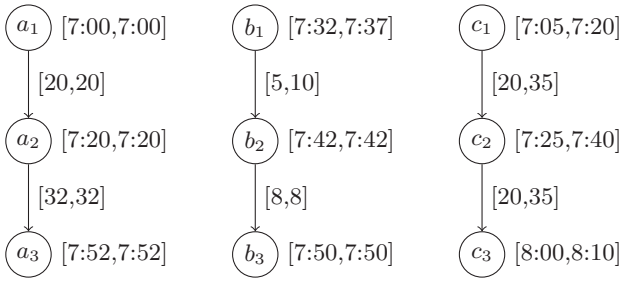

Figure 3: The optimal decoupling of the example STN

The valuation functions of the agents in Example 2 are all linear. This thus allows us to find an optimal decoupling of the STN given in that example in polynomial time using the LP translation given in this section.

Example 3. Translating the example on the agents controlling chemical processes (Example 2), we obtain an LP formulation with a 100-dimensional variable vector $\mathbf{p}$ and a constraint matrix containing 1000 TRI inequalities, 10 CON inequalities, 26 ORI inequalities, and 9 DEC inequalities. Note that some of these numbers are lower than expected at first glance; this is because we omit trivial inequalities featuring infinity.

The optimal solution to this problem (see Figure 3) has a total value of \$934 (which is, obviously, better than the decoupling in Figure 2 with total worth \$475), i.e. \$524 for agent A, \$10 for agent B and \$400 for agent C.

8. A MECHANISM FOR DECOUPLING

As we have shown above (see e.g. Example 2), in most settings many alternative decouplings are possible. An extreme example of a decoupling is when everything is fixed in advance, i.e. the constraints allow for only one solution. In other decouplings, some of the agents may not receive any additional constraints. However, this usually comes at the expense of other agents, which are then very restricted in their autonomy. Each agent usually has its own valuation for these alternatives. Sometimes, the values of the possible choices are private to the agents. Solving problems where essential information is private to self-interested agents creates the difficulty that some of these agents may try to manipulate the report of their private information in order to arrive at a solution that has more value to them. Therefore, we now show how to apply some known results from mechanism design to deal with this issue of manipulation in the context of decoupling. We refer the reader to e.g. [19] for a detailed exposition of the mechanism design concepts used below.

In this work, we focus on a so-called direct mechanism where we ask each player i to report a valuation function v_i that assigns a value to each possible decoupling. The set of all possible reports is called the set of strategies. If the dominant strategy for every player is to submit their correct valuation function, we say the mechanism is incentive compatible. Clearly, this is a desirable property when we are interested in optimizing the social welfare.

Let us now extend our model of OTD to include the players' valuation functions. We propose a valuation function v_i for each player i that takes as input the decoupled STP

instance, and as output some value in $\mathbb{R}$. We assume that v_i is constant in all constraint weights which are not contained inside $\mathcal{S}_i$. This is the (commonly used) no-externalities assumption; we assume here that no agent cares about the constraints in some other agent's subnetwork. In addition to the valuation, we allow a player's utility to also depend on some payment, i.e. $u_i(\{\mathcal{S}_k\}_{k=1}^N) = v_i(\{\mathcal{S}_k\}_{k=1}^N) - p_i(\{\mathcal{S}_k\}_{k=1}^N)$. When such payments are possible, this opens the way for the ubiquitous class of Groves mechanisms [11], which is the main successful class of mechanisms that optimize social welfare while satisfying incentive compatibility.

A Groves mechanism for OTD selects the optimal decoupling based on the declared valuations as the outcome. The payment, then, is given by $p_i(\{\mathcal{S}_k\}_{k=1}^N) = h_i(v_{-i}) - \sum_{j \neq i} v_j(\{\mathcal{S}_k\}_{k=1}^N)$, where h_i can be any function that does not depend on v_i. A common choice for h_i is the so-called Clarke tax [1], which in general is $\max_{\mathcal{S}'} \sum_{j \neq i} v_j(\{\mathcal{S}'_k\}_{k=1}^N)$. The Clarke tax can be interpreted as the marginal cost of an agent, and is computed by considering the same setting without taking the valuation of agent i into account. The resulting mechanism is known to be efficient (i.e., maximizes the social welfare) and incentive compatible (in dominant strategies), see e.g. [19] for the proofs.

We illustrate such a mechanism by elaborating upon our example set in the chemical industry (Examples 2 and 3).

Example 4. A Groves mechanism chooses the optimal outcome, so in this respect nothing changes from the result given in Example 3. However, in the previous example we simply assumed that we knew the valuations of all agents. By applying payments, a Groves mechanism makes the truthful declaration of these valuation functions to a centre incentive compatible. In case of the Clarke tax, such payments can be computed efficiently using the same linear program, but with a different objective, leading to the following payments in this case.

$$\begin{array}{lcrcrcl} p_A & = & 475 & - & 410 & = & 65 \\ p_B & = & 929 & - & 924 & = & 5 \\ p_C & = & 601.5 & - & 534 & = & 67.5 \end{array}$$

9. DISCUSSION AND RELATED WORK

The single most important difficulty added by multiagent systems is the coordination of self-interested agents. This coordination problem has been defined as *managing dependencies* and occurs in many more situations than just in multiagent systems; see e.g. [14] for an interdisciplinary survey. Considering its omnipresence in multiagent systems, the problem of analyzing this general coordination problem theoretically has received relatively little attention. Admittedly, there are many protocols and algorithms for coordination in specific settings, both on-line and off-line, but regarding a general model and analysis of the general coordination problem we have found only some work on social laws by Shoham and Tennenholtz et al. [9, 20, 21] and more recently work by Witteveen et al. [23]. It seems that social laws are not necessarily problem instance specific, while decouplings usually are, but nonetheless, technically these two concepts are essentially the same. Our work differs from all of the above mainly in that we study the decoupling problem in the context of multiple agents, generalizing the commonly used objective of (locally) maximizing flexibility to maximizing the social welfare. In addition, we show how to apply theory on VCG-based mechanisms to use our results in the context of self-interested rational agents that have private information. In contrast to social laws, such a chosen decoupling then does not need to be enforced upon these agents.

As an alternative to finding a socially optimal decoupling *before* execution, a number of techniques have been proposed to coordinate *during* execution, such as (Generalized) Partial Global Planning [6, 7, 8]. Some of these methods may be computationally more efficient than decoupling; in general, however, they only find locally optimal solutions because of the lack of look-ahead when making coordination decisions. Moreover, once a decoupling has been found, no more coordination (i.e., computational effort and communication) is required at all during execution. Thus, when taking only the execution phase into account, decoupling is more efficient than on-line methods; furthermore, one can envisage settings in which communication during execution is prohibitively expensive or simply impossible, making decoupling the only viable option. On the other hand, an advantage of on-line coordination approaches is that they may be able to deal with unforeseen changes more easily. As part of our future work, therefore, we will consider mixed forms based on both decoupling as well as on-line coordination.

Nisan and Ronen [17] show that sub-optimal truthful VCG-based mechanisms for cost minimization problems can lead to arbitrarily bad solutions. Their result also applies to optimal decoupling. This underlines the importance of finding optimal solutions when designing VCG-based mechanisms that are truthful. This aim for optimality led us to the following contributions, presented in this paper.

First, we have proved that finding decouplings for arbitrary distributed constraint systems is as hard as solving the constraint system centrally. From this, it immediately follows that *optimal* decoupling is at least as hard, and that if we aim for efficient algorithms for optimal decoupling, we should concentrate our attention on problems that are efficiently solvable. In this paper we therefore focused on the Simple Temporal Problem (STP), which is known to be tractable. We then solved the open problem put forward by Hunsberger regarding the complexity of finding an Optimal Temporal Decoupling (OTD) for the STP [13]. Surprisingly, OTD was shown to be intractable even for some quadratic objectives, but, fortunately, we were able to show that OTD is efficiently solvable when the objectives are linear, obtaining a socially optimal decoupling instead of a local optimum such as found by Hunsberger's algorithm. Finally, we have illustrated how an optimal algorithm for decoupling can be combined with known results from mechanism design, obtaining a mechanism that resolves coordination conflicts for rational, self-interested agents.

We believe that the mechanism design view on solving coordination problems between self-interested agents put forward in this paper can be an inspiration to continue work on social laws and decoupling. In particular, we expect significant contributions by studying other special cases like OTD that are polynomially solvable. Additionally, we leave for future work the study of decoupling for multiagent problems that are even harder than constraint satisfaction, such as e.g. multiagent planning, including nontemporal or mixed problem settings. In general, we cannot expect to be able to optimally decouple such intractable problems. This thus encourages us to consider recent work on for example second-chance or maximal-in-its-range mechanisms [17] to be able to

use approximation algorithms in the context of self-interested agents. In addition, in this work we assumed that coordination during execution is valued lower than any possible decoupling that can be obtained before execution and that agents therefore always meet the constraints posed by a decoupling. However, this cannot always be guaranteed, and especially when agents' valuations may depend on activities of other agents, this requires a two-stage mechanism [15, 18]. Taking such execution incentives into account will make our results on optimal decoupling applicable to an even wider range of domains.

10. REFERENCES

[1] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971.

[2] D. Cohen and P. Jeavons. *The complexity of constraint languages*, chapter 8. Elsevier, 2006.

[3] R. Dechter. *Constraint Processing*. Morgan Kaufmann Publishers, 2003.

[4] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artificial Intelligence*, 49(1–3):61–95, 1991.

[5] R. Dechter and J. Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38(3):353–366, 1989.

[6] K. S. Decker and V. R. Lesser. Generalizing the partial global planning algorithm. *International Journal of Intelligent and Cooperative Information Systems*, 1(2):319–346, June 1992.

[7] K. S. Decker and V. R. Lesser. Designing a family of coordination algorithms. In *Proceedings of the Thirteenth International Workshop on Distributed Artificial Intelligence (DAI-94)*, pages 65–84, 1994.

[8] E. H. Durfee and V. R. Lesser. Planning coordinated actions in dynamic domains. In *Proceedings of the DARPA Knowledge-Based Planning Workshop*, pages 18.1–18.10, Dec. 1987.

[9] D. Fitoussi and M. Tennenholtz. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119(1-2):61–102, 2000.

[10] G. Gottlob, N. Leone, and F. Scarcello. A comparison of structural CSP decomposition methods. *Artificial Intelligence*, 124(2):243–282, 2000.

[11] T. Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[12] L. Hunsberger. Algorithms for a temporal decoupling problem in multi-agent planning. In *Proc. of the National Conference on Artificial Intelligence*, pages 468–475. AAAI Press, 2002.

[13] L. Hunsberger. *Group decision making and temporal reasoning*. PhD thesis, Harvard University Cambridge, Massachusetts, 2002.

[14] T. W. Malone and K. Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, 21(1):87–119, 1994.

[15] C. Mezzetti. Mechanism design with interdependent valuations: Efficiency. *Econometrica*, 72(5):1617–1626, 2004.

[16] W. Naanaa. A domain decomposition algorithm for constraint satisfaction. *ACM Journal of Experimental Algorithmics*, 13:1.13–1.23, 2009.

[17] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. *Journal of AI Research*, 29:19–47, 2007.

[18] R. Porter, A. Ronen, Y. Shoham, and M. Tennenholtz. Fault tolerant mechanism design. *Artificial Intelligence*, 172(15):1783–1799, 2008.

[19] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-theoretic, and Logical Foundations*. Cambridge University Press, 2008.

[20] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artificial Intelligence*, 73(1-2):231–252, 1995.

[21] M. Tennenholtz. On social constraints for rational agents. *Computational Intelligence*, 15(4):367–383, 1999.

[22] B. W. Wah and Y. Chen. Constraint partitioning in penalty formulations for solving temporal planning problems. *Artificial Intelligence*, 170(3):187–231, 2006.

[23] C. Witteveen, W. van der Hoek, and N. Roos. Concurrently decomposable constraint systems. In *7th German Conference on Multiagent System Technologies*, volume 5774 of *Lecture Notes in Computer Science*, pages 153–164. Springer, 2009.

[24] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–685, 1998.