Distributed Decoupling Of Multiagent Simple Temporal Problems

Jayanth Krishna Mogali, Stephen F. Smith, Zachary B. Rubinstein
The Robotics Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
{jmogali, sfs, zbr}@cs.cmu.edu

Abstract

We propose a new distributed algorithm for decoupling the Multiagent Simple Temporal Network (MaSTN) problem. The agents cooperatively decouple the MaSTN while simultaneously optimizing a sum of concave objectives local to each agent. Several schedule flexibility measures are applicable in this framework. We pose the MaSTN decoupling problem as a distributed convex optimization problem subject to constraints having a block angular structure; we adapt existing variants of Alternating Direction Method of Multiplier (ADMM) type methods to perform decoupling optimally. The resulting algorithm is an iterative procedure that is guaranteed to converge. Communication only takes place between agents with temporal interdependences and the information exchanged between them is carried out in a privacy preserving manner. We present experimental results for the proposed method on problems of varying sizes, and demonstrate its effectiveness in terms of solving quality and computational cost.

1 Introduction

One basic challenge in many multiagent planning and scheduling domains is that of managing interdependencies between the planned activities of different agents over time. Effective, optimized multiagent schedules often require tight coordination among agents, but such schedules also tend to be brittle in dynamic and uncertain execution environments. To hedge against this problem, so-called "flexible times" planning and scheduling procedures (e.g., [Smith and Cheng, 1993; Laborie and Ghallab, 1995; Smith *et al.*, 2000; Policella *et al.*, 2009]) have been increasingly adopted as a basic means of retaining temporal flexibility in the plan. These procedures rely on an underlying representation of the plan/schedule as a Simple Temporal Network (STN) [Dechter *et al.*, 1991], an edge weighted graph that encodes all constraints between start and end points of activities in the plan and enables efficient determination of feasible activity execution intervals through temporal constraint propagation. In distributed planning settings, however, where each agent maintains a local STN that represents its portion of the overall multiagent plan, there is the additional complication of maintaining consistency across agent schedules as local planning constraints change during execution. If communications are reliable, then it is possible to maintain consistency by simply distributing the constraint propagation process [Smith *et al.*, 2007]. However in cases where communications are unreliable, a better solution might be to compromise somewhat on flexibility and decouple the subplans of constituent agents.

The concept of temporal decoupling was first introduced in [Hunsberger, 2002]. In brief, it entails elimination of all inter-agent temporal constraints (i.e., all constraints relating time points in the local STNs of two agents). For each such constraint, the set of feasible time point assignments admitted by this constraint is partitioned into two consecutive subintervals, and then new local constraints are substituted that allocate these subintervals respectively to both agents (The position where the interval of feasible assignments is split is sometimes referred to as the decoupling point.) Once a temporal decoupling is obtained, an individual agent is free to adjust its local schedule within the stated local bounds of its STN without disrupting the consistency of the overall joint multiagent plan. Initial centralized algorithms for computing a temporal decoupling were proposed in [Hunsberger, 2002], and in [Planken *et al.*, 2010] the tractability of the optimal problem was first established. Subsequently, [Boerkoel and Durfee, 2011; 2013] formulated temporal decoupling as a distributed multiagent problem, and proposed a distributed algorithm for decoupling an interdependent multiagent schedule (however with no guarantee of optimality). Other recent work [Wilson *et al.*, 2013; Witteveen *et al.*, 2014] has focused alternatively on developing more accurate characterizations of flexibility, which provide a stronger basis for optimization.

One advantage of a distributed approach to decoupling in several domains is its potential for sensitivity to privacy concerns. Consider a civil construction project, where different subcontractors are engaged to perform specific construction tasks (e.g., plumbing, painting). These contractors need to cooperatively construct a plan for the project subject to temporal dependencies (e.g., walls must be plastered before painted). At the same time, each subcontractor has its own private constraints (e.g., other contracting commitments and schedules), which would not normally be revealed to other contractors for business reasons. Decoupling the project plan gives in-

dividual contractors the flexibility to locally reorganize work plans to better accommodate new commitments without need for renegotiation with other contractors.

In this paper, we describe and analyze a new distributed algorithm for temporal decoupling that explicitly addresses agent privacy concerns. Our approach, which adapts recently proposed Alternating Direction Method of Multiplier (ADMM) type methods (e.g., [Boyd *et al.*, 2011]), is designed to share only the minimal amount of information about inter-agent dependencies necessary to optimally decouple, and thus agent privacy is preserved to the maximal extent possible. Our approach also offers several other advantages over previously developed decoupling algorithms. First, it provides a distributed decoupling procedure that, in contrast to the heuristic procedure of [Boerkoel and Durfee, 2013], is guaranteed to produce the optimal decoupling for a given decoupling objective, provided that the objective is a concave function. Second, by supporting concave objectives, the algorithm is not bound to a single definition of flexibility but can operate with several (including those specified in [Boerkoel and Durfee, 2013; Wilson *et al.*, 2013; Witteveen *et al.*, 2014] which are all linear). Third, it is even possible to ascribe different decoupling objectives to individual agents in circumstances where agents may be more self-interested and have different goals.

The remainder of the paper is organized as follows. First we summarize the multiagent decoupling problem. Next, we present our ADMM-based decoupling algorithm. We analyze the performance of our ADMM-based approach on the set of reference problems first introduced in [Boerkoel and Durfee, 2013] using their flexibility metric. By also solving these problems separately with a centralized optimization method, we show that in practice, our iterative ADMM procedure converges quickly to near-optimal solutions. We next analyze the algorithm's practical solving characteristics and communication requirements. Finally, we draw some conclusions and identify some directions for future research.

2 Temporal Decoupling Problem

The algorithm presented in this paper solves the distributed multiagent simple temporal network (MaSTN) decoupling problem first posed in [Boerkoel and Durfee, 2013]. We begin with a brief review of MaSTNs and then explain the MaSTN decoupling problem.

2.1 Multiagent Simple Temporal Network

Within a MaSTN, each agent's schedule is represented as a Simple Temporal Network (STN) and agent schedules form components of the overal MaSTN. An STN [Dechter *et al.*, 1991; Hunsberger, 2002] is a collection of time points $\mathcal{T} = [z, t_1, .., t_N]$ and a set of binary constraints $\mathcal{C}$ of the form $t_i - t_j \leq c_{ij}$, where $t_i, t_j \in \mathcal{T}$ and z is a fixed arbitrary time point reference. Following the convention of [Hunsberger, 2002], we represent an STN by the pair $< \mathcal{T}, \mathcal{C} >$. A feasible solution is an assignment of values to the elements in $\mathcal{T}$ that respects the constraints in $\mathcal{C}$. In [Dechter *et al.*, 1991] polynomial time algorithms are provided for computing such feasible assignments.

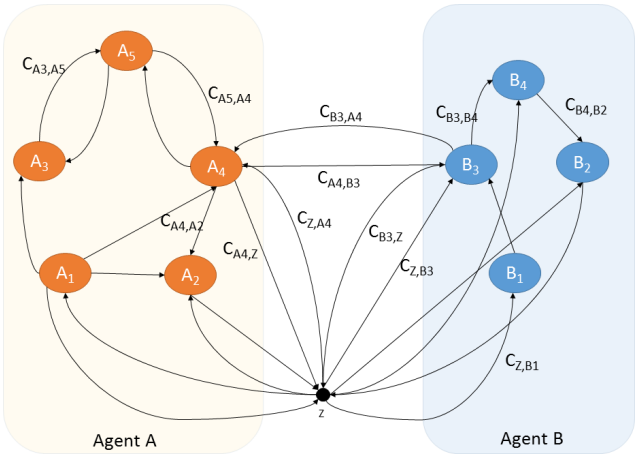

Figure 1: Decoupling a 2 agent MaSTN.

In the multiagent settings considered in this paper, it is assumed that each agent maintains its own private STN. Accordingly, we associate an agent with each $t_i \in \mathcal{T} \backslash \{z\}$ as its owner $O(t_i)$, and distinguish two types of constraints within the set $\mathcal{C}$ associated with the overall MaSTN: (1) intra-agent constraints, which are local to a given agent and considered private (i.e., $O(t_i) = O(t_j)$ or one of t_i, t_j is z), and (2) inter-agent constraints, which couple the STNs of different agents (i.e., $O(t_i) \neq O(t_j)$).

2.2 Decoupling

The overall goal of decoupling is to decompose the MaSTN into component STNs, one for each agent, that can be independently maintained with assurance that the overall MaSTN remains globally consistent. We explain the basic problem in two steps with the aid of the simple two-agent example shown in Fig.(1). In this example, we assume the following notation and definitions:

n_A → Number of time points owned by agent A+1
A_i → Time point i in agent A , $i \in \{1, \ldots, n_A\}$
T_A → $\{A_i\}_{i=1}^{n_A}$
n_B → Number of time points owned by agent B+1
B_i → Time point i in agent B , $i \in \{1, \ldots, n_B\}$
T_B → $\{B_i\}_{i=1}^{n_B}$
z → Common reference time point for all agents, $T_A \cap T_B = \{z\}$
C_{V_i,V_j} → Problem specified constraint value for edge from time point V_i to time point V_j where $V_i, V_j \in T_A \cup T_B$, and $V_i \neq V_j$.

We partition the constraint set $\mathcal{C}$ into inter and intra-agent constraints. The constraint linking time points A_3 to A_5 is an example of an intra-agent constraint and represented as a directed edge whose edge weight is given by C_{A_3,A_5}. Similarly, the edge connecting time points A_4 and B_3, is an example of an inter-agent constraint. The constraint set $\mathcal{C}$ is specified in the problem and is equivalently represented in the STN framework. For example, if the problem specifies the following constraints:

$$\begin{aligned} A_4 - A_2 &\leq 120 \\ A_2 - A_4 &\leq 60 \end{aligned}$$

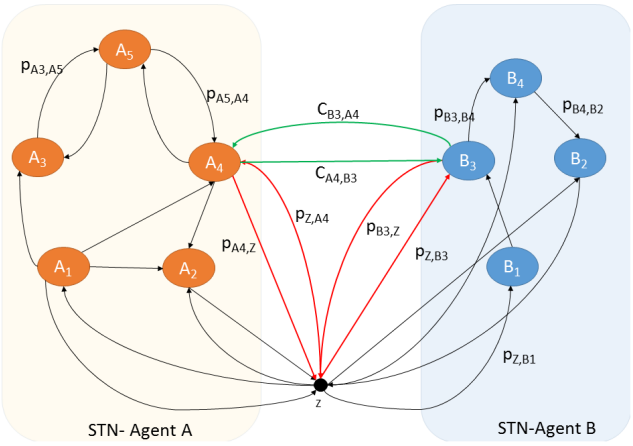

Figure 2: Corresponding minimal STN for problem in Fig 1. Green edges are inter-agent coupling edges. Red edges are tightened to dominate green edges.

then, we assign the values $C_{A_2,A_4} = 120$ and $C_{A_4,A_2} = -60$.

Using just the intra-agent constraints specified for each agent, agents A and B can separately compute STNs that are minimal with respect to intra-agent constraints [Dechter *et al.*, 1991]. All pairs of distinct vertices within the computed STNs are now connected through a directed edge. We denote the edge connecting vertices i and j for agent A's minimal STN with respect to intra-agent constraints as w_{A_i,A_j}. Intuitively the interval $[-w_{z,A_i}, w_{A_i,z}]$, represents the bounds on the value that time point A_i can be assigned in any feasible schedule subject to just the intra agent constraints. We denote a vector containing all the edges in the just computed minimal STN of agent A as $\mathbf{W_A} = [\{w_{A_i,A_j}\}], \forall i,j \in \{1,\ldots,n_A\}, i \neq j$, i.e. $\mathbf{W_A} = [w_{A_1,A_2}, w_{A_1,A_3}, \ldots, w_{A_1,A_{n_A}}, w_{A_2,A_1}, w_{A_2,A_3}, \ldots, w_{A_2,A_{n_A}}, \ldots, w_{A_{n_A},A_{n_A-1}}]^T$. We are now ready to formulate our two agent decoupling problem.

Constraints

The central idea behind decoupling is to tighten the bounds on time points that are involved in inter-agent constraints such that any assignment of values to those time points within those bounds can be decided independently by an agent without need for coordination with other agents participating in inter-agent constraints, while also ensuring such an assignment is feasible with respect to an agent's own intra-agent constraints. The approach we adopt is to start from the minimal STNs computed using just intra-agent constraints (i.e., $\mathbf{W_A}$ for each agent $\mathbf{A}$), and then let the agents cooperatively further tighten their corresponding STN edges so that the inter agent constraints are also implied. We denote the vector containing edges in the further tightened STN of agent A as $\mathbf{P_A} = [\{p_{A_i,A_j}\}], \forall i,j \in \{1,\ldots,n_A\}, i \neq j$ (refer to the definition of $\mathbf{W_A}$ above for expansion). Next, we present necessary constraints on $\mathbf{P_A}, \mathbf{P_B}$ that the agents need to satisfy for decoupling.

Since time points A_4 and B_3 in our problem in Fig.2 participate in an inter-agent constraint, decoupling the time points for that constraint yields the following decoupling equations:

$$(DEC)$$
$$p_{A_4,z} + p_{z,B_3} \leq C_{A_4,B_3} \quad (1)$$
$$p_{z,A_4} + p_{B_3,z} \leq C_{B_3,A_4} \quad (2)$$

For the inequalities that follow we present them for agent A, it is analogous for agent B. We also remind the reader that a minimal STN can be interpreted as a shortest path distance graph. Hence, all edges in the tightened STN for each agent must obey triangle inequality.

$$(TRI)$$
$$p_{A_i,A_j} - p_{A_i,A_k} - p_{A_k,A_j} \leq 0, \forall i,j,k \in \{1,\ldots,n_A\} \quad (3)$$

where $i \neq j \neq k$. While Eqn.(3) ensures that every closed path of length 3 on the STN is consistent, the following inequalities ensure that every path of length 2 is also consistent.

$$(CON)$$
$$-p_{A_i,A_j} - p_{A_j,A_i} \leq 0, \forall i,j \in \{1,\ldots,n_A\}, \& \, i > j \quad (4)$$

The upper bound value for each edge p_{A_i,A_j} is derived by noting that the decoupling process only shortens distances (edges) on the distance graph. For any valid p_{A_i,A_j}, we have

$$(UB)$$
$$p_{A_i,A_j} \leq w_{A_i,A_j} \quad (5)$$

The specification of all aforementioned constraints was previously developed in [Planken *et al.*, 2010]. To this formulation we introduce a way for computing the lower bounds for p_{A_i,A_j}. The sum of the edge weights along any closed path of lengths 2 and 3 must necessarily be ≥ 0 on any shortest path distance graph. For any valid p_{A_i,A_j}, we have:

$$(LB)$$
$$\begin{aligned} p_{A_i,A_j} &\geq max(-p_{A_j,A_i}, -p_{A_i,A_k} - p_{A_k,A_j}) \\ &\quad \forall k \in \{1,\ldots,n_A\}\backslash\{i,j\} \\ &\geq max(-w_{A_j,A_i}, -w_{A_i,A_k} - w_{A_k,A_j}), \\ &\quad \forall k \in \{1,\ldots,n_A\}\backslash\{i,j\} \end{aligned} \quad (6)$$

where Eqn.(6) is obtained from Eqn.(5). The LB constraints proposed are not necessary but can significantly improve the efficiency of the solving procedure (see Sec.3 below).

Optimal Decoupling

As the constraints defined in the previous section are all affine, a number of decoupling solutions can exist for a given MaSTN, and our objective in this paper is to find the optimal decoupling. We assume a proper concave function $(Flex)$ for defining the flexibility of a STN and parametrize it by the STN's edge weights. As the decoupling process decomposes the MaSTN into individual STNs, one for each agent, we define the social welfare as the sum of the flexibilities of each agent's STN. By optimal decoupling, we aim to maximize the social welfare. Previously developed flexibility measures (e.g., [Boerkoel and Durfee, 2013; Wilson *et al.*, 2013]) are all applicable in this framework. The resulting objective has a separable form with respect to the agents and hence is particularly suitable in applications that require minimal communication between agents and/or where certain privacy requirements exist.

Privacy Considerations

As indicated earlier, the time points and constraints in an agent's STN can be partitioned into private and shared variables. In Fig.(2), A_4 is a shared time point while elements in the set $\{T_A - A_4\}$ are private to agent A. For the problem in Fig.(2), Eqns.(1) and (2) suggest that the minimal information that agent B requires from agent A are the edges $p_{A4,z}$ and $p_{z,A4}$. Similarly, agent A's inference over the time points and constraints in agent B should be restricted to $p_{B3,z}$ and $p_{z,B3}$. Let us consider a case where we add a third agent C to the problem in Fig.(2) and include an inter-agent constraint between time points A_3 and C_2. Although, A_4 and A_3 are both shared variables, since agent C is coupled with agent A only via time point A_3, the decoupling algorithm should be able to restrict $C's$ inference over agent $A's$ variables to just $p_{A3,z}$ and $p_{z,A3}$. This minimal amount of information sharing specification in the problem has the additional advantage of minimizing the communication overhead between agents. The algorithm proposed in the next section produces the optimal MaSTN decoupling under these privacy considerations.

3 A Distributed Convex Optimization Representation For Decoupling

In this section, we represent the N agent decoupling problem in a convex optimization framework. Following the convention introduced in the previous section, we denote $\mathbf{P_i}$ as the vector of all edges in the STN of agent i and the problem is to find an optimal assignment of values to $\mathbf{P_i}, \forall i \in \{1, \ldots, N\}$ subject to the DEC, TRI, CON , LB and UB constraints. The decoupling problem in matrix notation is represented as:

Primal

$$\min_{\mathbf{P_1},\ldots,\mathbf{P_N}} \sum_{i=1}^{N} -Flex(\mathbf{P_i})$$

s.t.

$$\begin{bmatrix} E_1 & E_2 & . & . & E_N \\ D_1 & & & & \\ & D_2 & & & \\ & & . & & \\ & & & . & \\ & & & & D_N \end{bmatrix} \begin{bmatrix} \mathbf{P_1} \\ \mathbf{P_2} \\ . \\ . \\ \mathbf{P_N} \end{bmatrix} \leq \begin{bmatrix} b \\ c_1 \\ c_2 \\ . \\ c_N \end{bmatrix}$$

$$L_1 \leq \mathbf{P_1} \leq U_1, \ \ldots, \ L_N \leq \mathbf{P_N} \leq U_N$$

where,

$\mathbf{P_i} \rightarrow$ Vector of edge weights in agent $i's$ STN

$\sum_{i=1}^{N} E_i\mathbf{P_i} \leq b \rightarrow$ DEC constraints of all agents

$D_i\mathbf{P_i} \leq c_i \rightarrow$ TRI and CON constraints for agent i

$L_i \leq \mathbf{P_i} \leq U_i \rightarrow$ LB and UB constraints for $\mathbf{P_i}$

$[E] \rightarrow [E_1\, E_2\, \ldots\, E_N] \in \mathbb{R}^{W\times\cdot}$

Notice that the primal formulation is a minimization problem defined over a block diagonal constraint set. Matrices D_i and c_i for agent i is obtained by analogously representing all constraints in Eqns.(3) and (4) for agent A in matrix notation. All DEC constraints in the multi agent problem are collectively represented as $\sum_{i=1}^{N} E_i\mathbf{P_i} \leq b$, let the number of DEC constraints be W. We denote $E_{i,r}$ as a row vector and define it to be the entries in row r corresponding to agent $i's$ matrix E_i, since each row in E represents a DEC constraint between exactly 2 agents, entries in $E_{i,r}$ corresponding to non-interacting agents of row r are all set to 0. We first show how the primal problem is generically solved and later present the the necessary modifications to meet the privacy requirements in our problem. For the reminder of the paper, we convert DEC constraints into equality constraints through the addition of slack variables.[1]

Dual

$$\max_{\lambda} \ \sum_{i=1}^{N} \underbrace{\min_{\substack{D_i\mathbf{P_i}\leq c_i \\ L_i\leq\mathbf{P_i}\leq U_i}} -Flex(\mathbf{P_i}) + \lambda^T\left(E_i\mathbf{P_i} - b_i\right)}_{L_i(\lambda)}$$

$$\text{where } \sum_{i=1}^{N} b_i = b \tag{7}$$

To respect privacy, we set $b_i^r = \frac{b^r}{2}$ if $E_{i,r}$ is a non-zero vector, else $b_i^r = 0$, where b_i^r represents r^{th} row entry in vector b_i. Many distributed algorithms solve the primal problem by transforming to its dual. Every consistent MaSTN has a valid decoupling [Hunsberger, 2002] i.e. $\exists$ solution satisfying constraints in the primal; combining this fact with our concave flexibility metric assumption ensures that the primal and dual optimal objective values are equal, and the primal optimal solution is attainable from dual optimal solution [Boyd and Vandenberghe, 2004]. Hence we can equivalently work with the dual form since Eqn.(7) suggests that computations can be carried out in parallel by each agent. Solution strategies for the dual typically consists of an inner loop where each agent independently solves the minimization problem for the current estimate of the shared vector $\lambda \in \mathbb{R}^{W\times 1}$ and an outer loop where solutions from all agents are accumulated to update λ.

Recently, ADMM [Boyd *et al.*, 2011] type methods have been popularized as having improved convergence properties over prior art in terms of convergence with respect to primal variables $\mathbf{P_i}$ in Eqn.(7). The original ADMM algorithm takes a sequential approach to solving Eqn.(7) where each agent take turns in updating its variables. This limitation has been addressed in Consensus ADMM type methods such as the Dual Consensus ADMM (DC-ADMM) method [Chang *et al.*, 2015]. DC-ADMM is attractive in our context since it helps improve agent utilization. The philosophy of DC-ADMM is to have each agent maintain a private copy of λ (possibly with different values at the start of the algorithm), and drive their private estimates of λ to consensus through neighbor wise

[1] Since exactly two agents participate in each row/constraint of matrix E, we arbitrarily assign ownership of the slack variable to one of the participating agents (say k) and append it to $\mathbf{P_k}$. We represent the new DEC constraints as $\sum_{i=1}^{N} E_i\mathbf{P_i} = b$.

consensus constraints such as Eqns.(8) and (9).

DC-ADMM Primal

$$\max_{\substack{\lambda_1,\ldots,\lambda_N \\ \{m_{ij}\}}} \sum_{k=1}^{N} L_i(\lambda_k)$$

s.t.

$$\lambda_i = m_{ij}, \forall j \in Neigh(i), \forall i \in \{1,\ldots,N\} \quad (8)$$

$$\lambda_j = m_{ij}, \forall j \in Neigh(i), \forall i \in \{1,\ldots,N\} \quad (9)$$

where, $\lambda_i \in \mathbb{R}^{W\times 1}$ is agent $i's$ estimate, $Neigh(i)$ is the set of agent $i's$ neighbors with whom i exchanges the vector λ_i and m_{ij} is a consensus slack variable $\in \mathbb{R}^{W\times 1}$. Notice that DC-ADMM Primal and the Dual problem in Eqn.(7) are equivalent. The DC-ADMM Primal problem is solved using the standard ADMM framework; we refer our readers to [Chang *et al.*, 2015] for details.

Algorithmically, at each iteration of DC-ADMM, agents exchange their estimate of the dual vector λ with neighbors and use it to update their own estimate (cf Algorithm 3 in [Chang *et al.*, 2015]). In our problem, if agent i does not participate in row r, then $E_{i,r} = 0$ and $b_i^r = 0$. This implies that r^{th} row entry of the dual vector is inconsequential to agent i. Also, for DC-ADMM to work correctly, agent i is required to transmit values to its neighbors that are inconsequential to itself but they may be required for its neighbors. In a certain sense, transmitting these inconsequential values via agent i can be considered as a privacy violation; it is susceptible to manipulation; and is also inefficient to store and transmit.

To avoid communication of non-essential and potentially private information, we adjust the DC-ADMM procedure to ensure that an agent creates Lagrange dual variables $\in \mathbb{R}$ only for rows in matrix E that an agent participates in. This also ensures correspondingly fewer consensus constraints in our formulation. We define $R_E(i)$ as a function that outputs a set containing indices of all rows in which agent i is involved in matrix E. The function $Neigh_E(i,r)$ outputs j if agent i shares the constraint with agent j in row r of matrix E. A lagrange dual variable $y_{i,r} \in \mathbb{R}$ is allocated by agent i iff $r \in R_E(i)$, likewise $y_{j,r}$ is allocated by agent j iff $r \in R_E(j)$. Hence the modified DC-ADMM method for the decoupling problem can be represented as:

Modified DC-ADMM Primal

$$\max_{\substack{\{y_{i,r}\} \\ \{u_{i,r}\}}} \sum_{i=1}^{N} \min_{\substack{D_i\mathbf{P_i}\le c_i \\ L_i \le \mathbf{P_i} \le U_i}} -Flex(\mathbf{P_i}) + \sum_{r\in R_E(i)} y_{i,r}\left(E_{i,r}\mathbf{P_i} - \frac{b^r}{2}\right)$$

s.t.

$$y_{i,r} = u_{i,r},\ \ y_{j,r} = u_{i,r},\ \text{where } j = Neigh_E(i,r)$$
$$\forall r \in R_E(i), \forall i \in \{1,\ldots,N\}$$

where $u_{i,r}$ is a consensus slack variable $\in \mathbb{R}$. In Algorithm 1, we applied Sion's Minimax theorem and standard ADMM updates to the Modified DC-ADMM primal problem for solving. The steps taken to derive Algorithm 1 are identical to those taken in [Chang, 2014], so we only present the final algorithm (see Algorithm 1).

Algorithm 1 Modified DC-ADMM For Decoupling

Given: $c, k = 1, J_{i,r}^{(0)} = y_{i,r}^{(0)} = 0, \forall r \in R_E(i)$**, for each agent** $i \in \mathcal{V} = \{1\ldots N\}$

repeat

for $i \in \mathcal{V}$ in parallel **do**

$$\mathbf{P_i^{(k)}} = \arg \min_{\substack{D_i\mathbf{P_i}\le c_i \\ L_i \le \mathbf{P_i} \le U_i}} -Flex(\mathbf{P_i}) +$$

1:
$$\sum_{r\in R_E(i)} \frac{c}{4}\left(\frac{E_{i,r}\mathbf{P_i} - b_i^r - J_{i,r}^{(k-1)}}{c} + y_{i,r}^{(k-1)} + y_{j,r}^{(k-1)}\right)^2$$

where $j = Neigh_E(i,r)$

2: **for each** $r \in R_E(i)$ **do**

$j = Neigh_E(i,r)$

$y_{i,r}^{(k)} = \frac{1}{2c}(E_{i,r}\mathbf{P_i^{(k)}} - b_i^r - J_{i,r}^{(k-1)} + c(y_{i,r}^{(k-1)} + y_{j,r}^{(k-1)}))$

end for

3: **for each** $r \in R_E(i)$ **do**

$j = Neigh_E(i,r)$

$J_{i,r}^{(k)} = J_{i,r}^{(k-1)} + c(y_{i,r}^{(k)} - y_{j,r}^{(k)})$

end for

end for

k = k + 1

until Termination Criterion

At a high level, Algorithm 1 is executed in parallel by each agent with intermittent communication between them at each iteration. Notationally, we indicate the estimate of a variable (var) belonging to agent i at the k^{th} iteration as $var_{i,\cdot}^{(k)}$. The Algorithm accepts as input a penalty multiplier $c \in \mathbb{R}^+$. In step 1 of each iteration, all agents independently solve a minimization problem in polynomial time. The solution obtained $(\mathbf{P}_\cdot^{(\cdot)})$ will be feasible with respect to intra-agent constraints, with some infeasibility across DEC constraints. $\mathbf{P_i^{(k)}}$ corresponds to assignment of values to agent $i's$ STN at the k^{th} iteration. In step 2, the dual variables are updated using edge weights of the agent's STN computed in step 1 and relevant Lagrange dual values from neighbors in the previous iteration. The updated variables are then communicated to relevant neighbors. For example, if agents i and j share the constraint in the r^{th} row of matrix E, then agent $i's$ estimate of $y_{i,r}$ at the k^{th} iteration (i.e. $y_{i,r}^{(k)}$) is computed using $y_{j,r}^{(k-1)}$ and communicated only to agent j. Similarly, agent j updates and communicates $y_{j,r}^{(k)}$ only to agent i. Recall $R_E(i)$ outputs all relevant rows in matrix E for agent i and $Neigh_E(i,r)$ outputs the neighbor of agent i corresponding to row r iff $r \in R_E(i)$.

In practice however the agents do not jointly construct the matrix E (This has been introduced into Algorithm 1 to simplify the explanation.) The shared DEC constraints are assumed to be known by each respective pair of agents; the agents only need to ensure correct exchange of Lagrange variable values corresponding to each shared DEC constraint. In step 3, an agent's updated dual variables and those received from its neighbors are used to update temporary variables (for agent i this refers to variables belonging to the set $\{J_{i,r}\}_{\forall r\in R_E(i)}$) whose values are utilized in step 2. Note, the consensus slack variables appearing in the Modified DC-

ADMM primal problem were eliminated in the process leading to Algorithm 1.

3.1 Termination Criterion

We chose to use the extent of infeasibility of the DEC constraints as our termination criterion. If agents i and j share a DEC constraint at row r, it can be shown that $E_{i,r}\mathbf{P_i^{(k)}} + E_{j,r}\mathbf{P_j^{(k)}} - b^r = 2c(y_{i,r}^{(k)} + y_{j,r}^{(k)} - y_{i,r}^{(k-1)} - y_{j,r}^{(k-1)})$, where k is the iteration number. The term on the right side goes to zero as $k \to \infty$, owing to the convergence characteristics of ADMM. In our set up, the agents determine the magnitude of infeasibility for the DEC constraints at each iteration by simply examining the exchanged Lagrange dual values as the expression above suggests. When all DEC constraints are satisfied within an acceptable tolerance ϵ_{Inf}, Algorithm 1 terminates.

4 Empirical Analysis

4.1 Experiment Design

To evaluate the effectiveness of our approach, we tested our algorithm on a set of reference problems called the *BDH_Problem_Instances*[2], due to [Boerkoel and Durfee, 2013]. This dataset contains problems with varying numbers of agents, specifically there are 50 MaSTN instances each for 2, 4, 8, 12 , 16 , and 20 agent problems. In each problem, each agent has 10 tasks to complete, which implies each agent STN consists of 20 time points. The number of inter-agent constraints specified for each problem instance is $50 \times (N-1)$, where N is the number of agents in the problem.

The flexibility measure we optimized was that specified in [Boerkoel and Durfee, 2013] due to [Hunsberger, 2002], given that this was the assumed objective when this problem set was generated. To define flexibility below, we adopt the convention used in section 2.2. Mathematically, the flexibility between 2 time points A_i and A_j belonging to agent A is defined equal to $p_{A_i,A_j} + p_{A_j,A_i}$. The flexibility for agent A's STN is given by

$$Flex(\mathbf{P_A}) = \sum_{i,j \in \{1,\ldots,n_A\}, i>j} p_{A_i,A_j} + p_{A_j,A_i}$$

The objective we optimize is the sum of the flexibilities of all agents in our N agent decoupling problem. (We would expect to obtain comparable results to those presented below if the flexibility measure of [Wilson *et al.*, 2013] were substituted). The penalty multiplier c in Algorithm 1 was set to 1 on the assumption that the coupling constraints are fairly evenly distributed. The feasibility tolerance ϵ_{Inf} was set to 0.1 based on the judgment that a maximum infeasibility with regard to any single DEC constraint of $\leq$ one tenth of a time tick would be insignificant from a practical perspective.

Our ADMM-based decoupling algorithm was written in C++ with Message Passing Interface (MPI) for inter-agent communication. For performing minimization in Algorithm 1, we used an Interior Point Solver IPOPT [Wächter and Biegler, 2006] with warm start. Each problem instance was also solved centrally using IPOPT to provide optimal solutions for bench-marking. All experiments were carried out on an Intel 4 core i7-4790 processor at 3.6 GHz.

[2]https://data.3tu.nl/repository/uuid:ce3ad00b-d905-4be3-8785-228ae19e371a

Number Agents	Average Iterations	Average % Dev	Average Time (in sec)
2	209	1.59×10^{-3}	43
4	408.1	4.59×10^{-3}	122.6
8	595.9	5.98×10^{-4}	324.9
12	666.7	4.17×10^{-4}	591.3
16	658.2	3.105×10^{-4}	749
20	663.5	3.497×10^{-4}	940.5

Table 1: Convergence performance of proposed algorithm

4.2 Results

The experimental results are summarized in Table 1. For each problem set size, we show

- the average number of iterations until the feasibility tolerance threshold is met,
- the average % deviation of the solution found from the optimal solution, where % deviation is defined as:

$$\%\, Dev = \frac{OptimalFlex - ComputedFlex}{OptimalFlex} \times 100$$

- average total computation time required.

As can be seen, optimizing performance is quite good; the percentage deviation from the optimal ranging from .0016 to .0006 of 1% depending on the size of the problem. On feasibility, the termination condition ensures that the maximum infeasibility with regard to any one DEC constraint is 0.1 time tick. However, we have also examined the Root Mean Squared (RMS) error of infeasibility of constraints (indicative of the magnitude of individual DEC constraint violations), defined as

$$RMS_{Inf} = \left(\sum_{l=1}^{L} \frac{(l^{th}\, DEC\, Violation\, Magnitude)^2}{L} \right)^{\frac{1}{2}}$$

where L is the number of DEC constraints. When the values obtained for instances of certain problem size are averaged, the resulting average RMS values range from 0.01 to 0.06 depending on problem size. Hence the average infeasibility associated with a particular DEC constraint is closer to one hundredth of a time tick. Since [Boerkoel and Durfee, 2013] have not to our knowledge published the solutions they obtained on these problems, it is not possible to perform a direct performance comparison with their approach. However, we can note that their approach makes no claim of finding optimal solutions. The computation times shown in Table 1 were achieved using a single 4-core machine, which, for the 20-agents test problems, required multiple agents to share cores. In practice, where we expect agents to operate independently and have devoted cores, the computation time will significantly reduce. Moreover, as shown in the plot of a representative run in Fig.3, the algorithm quickly drops to the vicinity of

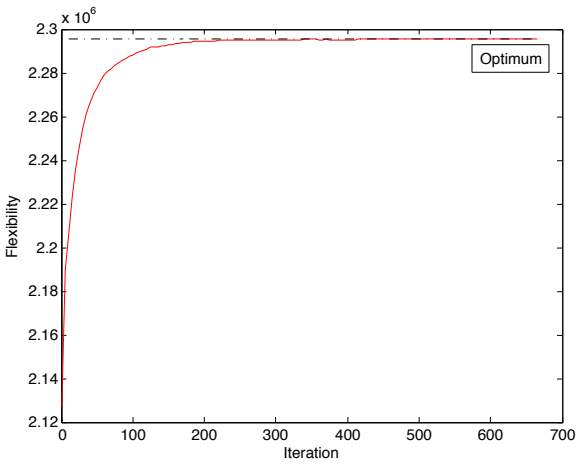

Figure 3: Solving progress on a 20 agent problem instance. Termination condition is satisfied at iteration 671.

the optimal and then oscillates around this neighborhood until the acceptable tolerance is achieved. If a compromise can be made on optimality such that a larger tolerance is acceptable, then the computation time will be substantially shorter.

Finally, a few words about message traffic. Lack of data on the amount of communication required by the approach of [Boerkoel and Durfee, 2013] on these problems again prevents a direct comparison along this dimension. However we can distinguish differences in communication requirements. If we consider a single Lagrange dual variable transmitted from one agent to another as the smallest bit of information, then the number of such bits transferred over the course of generating a decoupling can be seen to be $|DEC\,Constraints| \times 2 \times iterations$ (i.e., the product of the number of DEC constraints times twice the number of iterations it took for convergence). In comparison, the method of [Boerkoel and Durfee, 2013] entails establishment of a chordal graph and requires communication each time an edge is formed or updated which is of $O({V_s}^2)$, where V_s is the number of shared time points.

5 Conclusions And Future Work

In this paper, we have presented a new distributed algorithm for decoupling inter-dependent multiagent simple temporal networks (MaSTNs). We formulate the problem as a convex optimization problem and adapt a recently proposed Alternative Direction Method of Multiplier (ADMM) method to specify a distributed, iterative procedure that is guaranteed in the limit to converge to the optimum, given a concave flexibility objective. The procedure works by tolerating some amount of infeasibility across agent plans/schedules, and iteratively minimizing the magnitude of infeasibility (the dual objective) to some acceptable threshold. Experimental analysis of the procedure on a set of reference MaSTN decoupling problems has shown strong performance. Over this problem set, the solutions found are within .0016 - .0005 of 1% of the optimal decoupling, and convergence is observed to require a relatively small number of iterations.

Our ADMM-based decoupling procedure improves on prior work in decoupling of MaSTNs in several ways. Most importantly, it provides for the first time a decoupling algorithm that addresses agent privacy concerns, requiring sharing of only the minimal amount of information necessary to produce an optimal decoupling. It is also the first distributed decoupling algorithm offering an optimality guarantee. Finally, it accommodates a range of decoupling objectives, making the approach more broadly applicable to more self-interested settings where agents may have different objectives.

One potential limitation of the approach, given its reliance on a matrix formulation, is its scalability to large agent plans/schedules. An important point not made explicit in our formulation is that our objective metric requires all edge information, which prohibits the use of Partial Path Consistency (PPC) [Planken *et al.*, 2008] as a way of reducing the number of TRI constraints. One thrust of our current work aims at investigating other approaches to reducing model size.

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions in improving the presentation of this paper. The research reported in this paper has been sponsored in part by the Department of Defense Advanced Research Projects Agency (DARPA) under contract # FA8750-15-C-0137 , and the CMU Robotics Institute.

References

[Boerkoel and Durfee, 2011] James C. Boerkoel and Edmund H. Durfee. Distributed algorithms for solving the multiagent temporal decoupling problem. In *Proceedings 10th International Conference on Autonomous Agents and Multiagent Systems*, May 2011.

[Boerkoel and Durfee, 2013] James C. Boerkoel and Edmund H. Durfee. Distributed reasoning for multiagent simple temporal problems. *Journal of Artificial Intelligence Research*, 47:95–156, 2013.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Boyd *et al.*, 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[Chang *et al.*, 2015] Ting-Hau Chang, Mingyi Hong, and Xiongfei Wang. Multi-agent distributed optimization via inexact consensus admm. *Signal Processing, IEEE Transactions on*, 63(2):482–497, 2015.

[Chang, 2014] Tsung-Hui Chang. A proximal dual consensus admm method for multi-agent constrained optimization. *arXiv preprint arXiv:1409.3307*, 2014.

[Dechter *et al.*, 1991] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial Intelligence*, 49:61–95, May 1991.

[Hunsberger, 2002] Luke Hunsberger. Algorithms for a temporal decoupling problem in multi-agent planning. In *18th National Conf. on Artificial intelligence*, pages 468–475, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[Laborie and Ghallab, 1995] Philipe Laborie and Malik Ghallab. Planning with sharable resource constraints. In *Proceedings 1995 International Joint Conference on Artifical Intelligence*, pages 1643–1649, 1995.

[Planken *et al.*, 2008] Léon Planken, Mathijs De Weerdt, and Roman van der Krogt. P3c: A new algorithm for the simple temporal problem. In *Proceedings Eighteenth International Conference on Automated Planning and Scheduling*, pages 256–263, Sydney, Australia, 2008.

[Planken *et al.*, 2010] Léon R Planken, Mathijs M De Weerdt, and Cees Witteveen. Optimal temporal decoupling in multiagent systems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 789–796. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[Policella *et al.*, 2009] Nicola Policella, Amedeo Cesta, Angelo Oddi, and Stephen F. Smtih. Solve-and-robustify: Synthesizing partial order schedules by chaining. *Journal of Scheduling*, 12(3), 2009.

[Smith and Cheng, 1993] Stephen F. Smith and C. Cheng. Slack-based heuristics for constraint satisfaction scheduling. In *Proceedings 11th National Conference on Artificial Intelligence*, July 1993.

[Smith *et al.*, 2000] David E. Smith, Jeremy Frank, and Ari K. Jonsson. Bridging the gap between planning and scheduling. *Knowledge Engineering Review*, 15(1):47–83, Mar 2000.

[Smith *et al.*, 2007] Stephen F. Smith, Anthony T. Gallagher, Terry Lyle Zimmerman, Laura Barbulescu, and Zack Rubinstein. Distributed management of flexible times schedules. In *Proceedings 6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2007.

[Wächter and Biegler, 2006] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[Wilson *et al.*, 2013] Michel Wilson, Tomas Klos, Cees Witteveen, and Bob Huisman. Flexibility and decoupling in the simple temporal problem. In F. Rossi, editor, *Proceedings International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2422 – 2428. AAAI Press, Menlo Park, CA, 2013.

[Witteveen *et al.*, 2014] Cees Witteveen, Michel Wilson, and Tomas Klos. Optimal decoupling in linear constraint systems. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, pages 2381–2387. AAAI Press, Menlo Park, CA, August 2014.