

Quadratic Programming for Multi-Target Tracking

Raghav Aras and Alain Dutech and Francois Charpillet

Raghav.Aras@loria.fr, Alain.Dutech@loria.fr, Francois.Charpillet@loria.fr

MAIA - LORIA/INRIA

Campus Scientifique - BP 239

54506 Vandoeuvre les Nancy - France

MSDM Workshop, 2009

Abstract

We consider the problem of tracking multiple, partially observed targets using multiple sensors arranged in a given configuration. We model the problem as a special case of a (finite horizon) DEC-POMDP. We present a quadratic program whose globally optimal solution yields an optimal tracking joint policy, one that maximizes the expected targets detected over the given horizon. However, a globally optimal solution to the QP cannot always be found since the QP is nonconvex. To remedy this, we present two linearizations of the QP to equivalent 0-1 mixed integer linear programs (MIPs) whose optimal solutions, which may be always found through the branch and bound method, for example, yield optimal joint policies. Computational experience on different sensor configurations shows that finding an optimal joint policy by solving the proposed MIPs is much faster than using existing algorithms for the problem.

1 Introduction

This paper addresses a special case of finite horizon DEC-POMDPs. The special case has been called a network distributed POMDP [4] or a factored DEC-POMDP [5]. Lately, this special case has received attention in these pages, especially for the problem of detecting multiple targets passing through a given configuration of locations using multiple sensors, and specialized algorithms have been conceived for it [4], [6], [3]. Our focus too shall be on the multi-target tracking problem.

In this problem, the set of agents (sensors) is partitioned into subsets. It is assumed that for each subset, we can define immediate rewards that are dependent on the actions of the agents of the subset but not on the actions of agents outside the subset. It is furthermore assumed that the probabilities with which an agent receives observations are independent of probabilities with which other agents receive observations. Finally, it is assumed that the probabilities of transitions between states are independent of actions of the agents.

The purpose of the above partitioning scheme is to model autonomy for agents in one subset from those in other subsets. In the multi-target tracking problem (Figure 1), only the two sensors surrounding

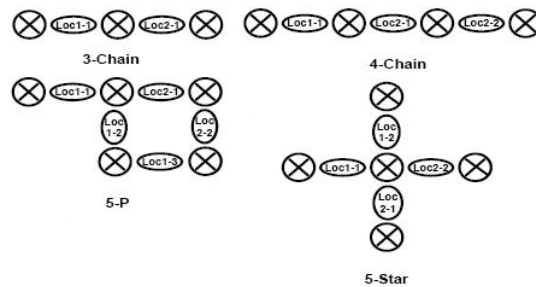


Figure 1: Sensor configurations (reprinted from [6]).

a location are required to detect a target at that location; the other sensors play no role in the target's detection. Each sensor enjoys autonomy from all the other sensors save its immediate neighbor(s) in any of the four cardinal directions. In this problem, state transition probabilities are independent of the sensors' actions since a sensor's choice of location to monitor (its action) does not influence the targets' positions (the state of the problem).

Recently, in [1], mathematical programming was applied with encouraging results to the general case of finite horizon DEC-POMDPs. Specifically, a nonlinear program (nonlinear objective, linear constraints) and 0-1 mixed integer linear programs (MIPs), equivalent to the nonlinear program, were presented. The use of the sequence form of a policy facilitated the conception of these programs. Exploiting the power of ILOG CPLEX and NEOS solvers, these programs were able to solve sample problems much more rapidly than other algorithms (by two orders of magnitude).

In this paper, we adapt this mathematical programming approach to the multi-target tracking problem. For the configurations of locations considered, the set of agents partitions into subsets that each contain exactly two agents. This being so, the adaptation of the nonlinear program yields a quadratic program. Following [1], we linearize this QP to a 0-1 MIP. While solving the QP is only guaranteed to find a locally optimal joint policy (but in practice often finds the optimal joint policy), the solving MIP is guaranteed to return an optimal joint policy. Secondly, we present a *new* linearization of the quadratic program to a 0-1 MIP. The new 0-1 MIP uses exponentially fewer variables and constraints than the other 0-1 MIP. This new MIP is in fact also usable for solving the general case of DEC-POMDPs. Computational experience of the programs on different location configurations reveals that the programs are much faster than existing approaches for the problem, the improvement in time being of the same order as for problems of general, unpartitioned DEC-POMDPs.

2 The Model

The special case of a DEC-POMDP is specified by the following data:

$I = \{1, 2, \dots, n\}$, a set of agents. S , a set of states. A_i and O_i , sets of respectively actions and observations of agent $i \in I$. For each pair of states $s, s' \in S$, the probability $p(s, s')$ that the process moves from s to s' , is defined. As stated earlier, we assume that this probability is not conditional on the actions of the agents. For each $i \in I$, for each $a \in A_i$, for each $s' \in S$, and for each $o \in O_i$, the probability $q_i(a_i, o_i, s')$ that i receives o in s' if he has taken a , is defined. Again, as stated earlier, we assume that this probability is not conditional on the observations or actions of other agents.

The set I of agents is partitioned into subsets which together exhaust it. The set of these subsets is denoted by D . Each subset in D has one or more agents in common with at least one other subset in D .

For each $d \in D$, let A^d denote the set $\times_{i \in d} A_i$ (joint actions over d). Thereby, for each $d \in D$, immediate rewards are defined: that is, for every $s \in S$ and for every joint action $d \in A^d$, the reward $R^d(s, a)$ of the agents of d taking a in s , is defined. Thereby, the total reward obtained by the agents in a period if the state is s and they take the joint action $a \in A$ is $\sum_{d \in D} R^d(s, a(d))$ where $a(d) \in A^d$ is the joint action over d formed by the elements of a .

Let T denote the horizon of the problem. Let Z_i denote the set of all possible sequences of $T - 1$ or less observations conceivable from O_i . Z_i also includes the null sequence. The *policy* of an agent is a function from Z_i to A_i . In using a policy π_i , the agent takes action $\pi_i(z)$ in a period if the sequence of observations he has received till that period is z . A *mixed* policy is a probability distribution over the set of policies. A *joint policy* is a n -tuple of policies, one policy in the tuple per agent.

Let $\Delta(S)$ denote the set of probability distributions over S . For $b^* \in \Delta(S)$, an *optimal joint policy* at b^* is a joint policy $\pi = (\pi_1, \pi_2, \dots, \pi_n)$ that maximizes the expectation of $\sum_{t=1}^T \sum_{d \in D} R^d(s^t, \pi^d(z^{d,t}))$ where s^t is a random variable representing the state in period t , $z^{d,t}$ is a random variable representing the tuple of sequences of observations received by the agents in d till period t , and $\pi^d(z^{d,t})$, the corresponding joint action according to π^d , the joint policy over d formed from π , and where s^1 is according to b^* .

2.1 Example

We follow the specifications of the multi-target tracking problem given in [4]. Consider the 4-chain configuration given in Figure 1. The four sensors in the chain together are meant to detect mobile objects (targets) that appear at the three locations following a fixed Markovian law. Two types of

targets are possible: targets of type 1 appear only at location Loc1-1 while targets of type 2 can appear at locations Loc2-1 or Loc2-2.

Each sensor is an agent. Sensors 1 and 2 are assigned to monitor location Loc1-1, sensors 2 and 3 are assigned to monitor location Loc2-1 while sensors 3 and 4 are assigned to monitor location Loc2-2. Thus, $I = \{1, 2, 3, 4\}$ and $D = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$.

In a period, a sensor can either monitor the location to its left (\supset) or monitor the location to its right (\subset) or switch itself off (\emptyset). A target is detected at a location only if the location is being monitored by the sensors to its left and right (or, in other configurations such as Five-P, by the sensors to its top and bottom); if only one of these two sensors is monitoring the location, the target slips by undetected. For instance, if a target is at location Loc1-1 in a period, it is detected only if sensor 1 chooses \subset and sensor 2 chooses \supset , and if a target is at location Loc2-2 in a period, it is detected only if sensor 2 chooses \subset and sensor 3 chooses \supset . If targets appear at these two locations in the same period, sensor 2 must decide which location is preferable to monitor in that period.

The state of the problem in a period is described by the a pair of numbers (x, y) with $x \in \{0, 1\}$ and $y \in \{0, 1, 2\}$. $x = 0$ denotes the absence of target 1 in Loc1-1 and $x = 1$ its presence in Loc1-1. $y = 0$ denotes the absence of target 2 in both Loc2-1 and Loc2-2, $y = 1$ its presence in Loc2-1 and $y = 2$ its presence in Loc2-2. There are thus 6 possible states in the problem. The state of the problem evolves from one period to another according to probabilities that are independent of the agents' actions.

A target arriving at a location is partially observed by the sensors assigned to the location and unobserved by the sensors not assigned to the location. The observations received by a sensor assigned to a location are '1' (for presence of a target) and '0' (for absence of target).

Finally, each monitoring action of a sensor in a period obtains a reward. The monitoring of a targetless location has a small negative reward. The detection of a target at a location results in a large positive reward obtained collectively by *all* the sensors. If a sensor switches itself off, there is no reward. Our objective in this problem is to maximize the expected number of targets detected in a given number of periods, and to do so we must maximize the total expected reward obtainable in those periods.

3 The Sequence Form

The *sequence form* of a policy is a representation of a (possibly, mixed) policy that facilitates formulating the problem of finding an optimal joint policy as a mathematical program. It was introduced in [2] for games in extensive form and used in [1] for finite horizon DEC-POMDPs.

The sequence form of a policy of an agent is defined as a conditional probability distribution over the set of histories of the agent. We define a history of length $t \geq 1$ of an agent to be a sequence of length $2t - 1$ in which the elements in odd positions are actions of the agent and those in even position, his observations. Thus, a history of length t has t actions and $t - 1$ observations. A history of length 1 is just an action. A history of length T shall be called *terminal*.

To give examples of histories from the 3 or 4-chain configuration of the multi-target tracking problem where $A_i = \{\subset, \supset, \emptyset\}$ and $O_i = \{0, 1\}$, $\subset-1-\supset-0-\subset$ is a history of length 3, $\supset-0-\supset$ is a history of length 2, \emptyset is a history of length 1 etc.

Let W_i denote the set of histories of lengths less than or equal to T of agent i . Let Y_i denote the set of terminal histories of agent i . A policy in the sequence form of agent i is a function σ_i from W_i to $[0, 1]$ such that,

$$\sum_{a \in A_i} \sigma_i(a_i) = 1 \quad (1)$$

$$\sigma_i(h) = \sum_{a \in A_i} \sigma_i(h, o, a), \quad \forall h \in W_i \setminus Y_i, \forall o \in O_i \quad (2)$$

where h, o, a denotes the history obtained on concatenating o and a to h . For history $h = (a^1, o^1, a^2, \dots, o^{t-1}, a^t)$ $\sigma_i(h)$ is the conditional probability,

$$\text{prob}(a^1, a^2, \dots, a^t | o^1, o^2, \dots, o^{t-1}) \quad (3)$$

In using a policy σ_i in the sequence form, the agent takes action a in period t upon receiving observation o in that period with probability $\sigma_i(h_t, o, a) / \sigma_i(h_t)$, where h_t is the history that has occurred till that period.

(Note: That any (mixed) policy π_i in the canonical form can be converted to its equivalent sequence form is self-evident. The converse can also be shown: For every policy σ_i in the sequence form, there exists a (possibly, mixed) policy π_i in the canonical form such that for each history $h \in W_i$, the conditional probability of the form (3) assigned to h by σ_i is the same as assigned by π_i .)

3.1 The Expected Reward

The expected reward of a joint policy (in the sequence form) can be expressed in terms of the sum of the expected rewards of the terminal joint histories of each subset $d \in D$. Denoting the size of d by m , a terminal joint history of d is an m -tuple of terminal histories, one history in the tuple per agent in d .

Let Y^d denote the set $\times_{i \in d} Y_i$ of terminal joint histories of d . In a joint history $J \in Y^d$, let J_i denote the history of agent $i \in d$ in h . Let $r^d(J)$ denote the expected reward of $J \in Y^d$. Then, the expected reward of a joint policy $(\sigma_1, \sigma_2, \dots, \sigma_n)$ in the sequence form is given by,

$$\sum_{d \in D} \sum_{J \in Y^d} r^d(J) \prod_{i \in d} \sigma_i(J_i) \quad (4)$$

Let O^d denote the set $\times_{i \in d} A_i$ (joint observations over d). For a terminal joint history $J = (a^1, o^1, a^2, o^2, \dots, o^{T-1}, a^T) \in Y^d$, where each a^k is a joint action in A^d and each o^k is a joint observation in O^d . Thereby,

$$r^d(J) = \left\{ \prod_{t=1}^{T-1} P(o^t | b^{d,t}, a^t) \right\} \left\{ \sum_{t=1}^T R^d(b^{d,t}, a^t) \right\}$$

The elements appearing in the r.h.s. are as follows. For $t = 1$, $b^{d,t} = b^*$; for $t > 1$, for each $s' \in S$,

$$b^{d,t}(s') = \sum_{s \in S} b^{d,t-1}(s) p(s, s') \prod_{i \in d} q_i(a_i^{t-1}, s', o_i^{t-1})$$

, for each $t \geq 1$,

$$R^d(b^{d,t}, a^t) = \sum_{s \in S} b^{d,t}(s) R^d(s, a^t)$$

and,

$$P(o^t | b^{d,t}, a^t) = \sum_{s \in S} b^{d,t-1}(s) p(s, s') \sum_{s' \in S} \prod_{i \in d} q_i(a_i^{t-1}, s', o_i^{t-1})$$

4 Quadratic Program

A policy in the sequence form is a solution to system of linear equations and inequalities. To be precise, a solution to the following system, based on (1)-(2), is a policy in the sequence form of agent i ,

$$\sum_{a \in A_i} x_i(a) = 1 \quad (5)$$

$$-x_i(h) + \sum_{a \in A_i} x_i(h, o, a) = 0, \quad \forall h \in W_i \setminus Y_i, \forall o \in O_i \quad (6)$$

$$x_i(h) \geq 0, \quad \forall h \in W_i \quad (7)$$

This system consists of one variable $x_i(h)$ for each history $h \in W_i$. Thus, the variable $x_i(h, o, a)$ is for the history obtained on concatenating o and a to the history h . Let the size of W_i be denoted by n_i . Let m_i denote the number of equations in (5)-(6). Let C_i denote an $m_i \times n_i$ matrix containing the coefficients of the left-hand sides of (5)-(6). Let c_i denote an m_i -vector containing the right-hand sides of (5)-(6); thus, the first entry of c_i is 1 and the remaining entries are 0s. Then, (5)-(7) can be written as $C_i x_i = c_i$, $x_i \geq \mathbf{0}$. Note that matrix C_i is sparse (most of its entries are 0s).

The set of policies X_i of agent i is a polyhedron.

$$X_i = \{x_i \in \mathbb{R}^{n_i} | C_i x_i = c_i, x_i \geq \mathbf{0}\}$$

A joint policy is an n -tuple of points, each point in a distinct polyhedron.

The discussion so far leads us directly to a linearly constrained quadratic program for the problem of multi-target tracking. In the multi-target tracking problem, in each configuration considered, there are only two agents in each subset d . Hence, the subsets in D can be numbered as 12, 23, 34 etc. (12 means that agents 1 and 2 are in subset 12). Therefore, the expected reward (4) of a joint policy $(\sigma_1, \sigma_2, \dots, \sigma_n)$ for this problem assumes a quadratic form. For example, for the 4-chain configuration, the expected reward is,

$$\begin{aligned} \sum_{J \in Y^{12}} r^{12}(J) \sigma_1(J_1) \sigma_2(J_2) + \sum_{J \in Y^{23}} r^{23}(J) \sigma_2(J_2) \sigma_3(J_3) \\ + \sum_{J \in Y^{34}} r^{34}(J) \sigma_3(J_3) \sigma_4(J_4) \end{aligned}$$

The expected reward of a joint policy can be expressed in matrix form as follows. Let the histories of each set W_i be numbered from 1 to n_i . For the 3-chain configuration, $D = \{\{1, 2\}, \{2, 3\}\}$. Define an $n_1 \times n_2$ matrix M^{12} whose rows are indexed by the histories of W_1 and whose columns by the histories of W_2 , and whose fg th entry is,

$$M_{fg}^{12} = \begin{cases} r^{12}(f, g), & \text{if } f \text{ and } g \text{ are both terminal histories} \\ 0, & \text{otherwise} \end{cases}$$

Define an $n_2 \times n_3$ matrix M^{23} analogous to M^{12} . Then, the expected reward of a joint policy $(\sigma_1, \sigma_2, \sigma_3)$ for this configuration is,

$$\sigma_1' M^{12} \sigma_2 + \sigma_2' M^{23} \sigma_3$$

σ' denotes the transpose of σ .

The expected reward of a joint policy for the other configurations can be similarly expressed in terms of matrices. For the 4-chain configuration, it can be expressed in terms of matrices M^{12} , M^{23} and M^{34} ; for the 4-star configuration, in terms of matrices M^{12} , M^{23} and M^{24} ; for the 5-star configuration, in terms of matrices M^{12} , M^{23} , M^{24} and M^{25} ; for the 5-P configuration, in terms of matrices M^{12} , M^{23} , M^{25} , M^{34} and M^{45} .

An optimal solution to the following quadratic program is an optimal joint policy for the 3-chain configuration,

$$\text{maximize} \quad x_1' M^{12} x_2 + x_2' M^{23} x_3$$

subject to,

$$x_i \in X_i, \quad i = 1, 2, 3$$

A globally optimal solution x_1^* , x_2^* , x_3^* to this QP is an optimal joint policy.

While this QP for the 3-chain configuration, its skeleton is applicable in fact to all the configurations of the problem. The only changes that are required to the program when moving from one configuration to another are to rewrite the objective function and to either add or remove sets of policy constraints (depending on whether the new configuration has more or less sensors than the previous configuration). For instance, an optimal solution to the following QP is an optimal joint policy for the 4-star configuration,

$$\text{maximize} \quad x_1' M^{12} x_2 + x_2' M^{23} x_3 + x_2' M^{24} x_4$$

subject to,

$$x_i \in X_i, \quad i = 1, 2, 3, 4$$

The general form of the QP for the multi-target tracking problem is,

(Q)

$$\text{maximize} \quad \sum_{d \in D} x'_i M^d x_{-i}$$

subject to,

$$x_i \in X_i, \quad i = 1, 2, \dots, n$$

where for a given $d \in D$, i and $-i$ represent respectively the indices of the two agents (sensors) that belong to $d \in D$.

Proposition 1. *A globally optimal solution $(x_1^*, x_2^*, \dots, x_n^*)$ to Q is an optimal joint policy.*

Proof. By definition of a policy in the sequence form and the expected reward of a joint policy in the sequence form. \square

As an algorithm, however, Q is not ideal because it is nonconvex (in most cases). In other words, in most cases, none of the matrices,

$$\begin{pmatrix} 0 & -M^d \\ -M^{d'} & 0 \end{pmatrix}$$

is positive semi-definite. Solving Q is thereby guaranteed to yield only a locally optimal joint policy. In the next section, we convert Q to equivalent 0-1 mixed integer linear programs, solving which is guaranteed to yield an optimal joint policy.

5 Mixed Integer Programs

As stated in the opening, we present two different 0-1 mixed integer linear programs (MIPs) that are equivalent to Q in the sense that an optimal solution to the MIP is also a globally optimal solution to Q.

Both MIPs are based on the linearization of the objective of Q. Both MIPs yield an optimal joint policy that is *pure*, that is one in which each policy assigns conditional probabilities to histories that are either 0 or 1. The first MIP was described in [1] while the second MIP is novel.

The 0-1 MIP due to [1] is as follows.

(M1)

$$\text{maximize} \quad \sum_{d \in D} m'_d z_d$$

subject to,

$$\begin{aligned} x_i &\in X_i, \quad \forall i \in I \\ l_{-i} x_i(h) &= \sum_{J \in Y^d: J_i=h} z_d(J), \quad \forall i \in I, \forall d \in D_i, \forall h \in Y_i \\ \sum_{J \in Y^d} z_d(J) &= l_i l_{-i}, \quad \forall d \in D \\ 0 \leq z_d(J) &\leq 1, \quad \forall d \in D, \forall J \in Y^d \\ x_i(h) &\in \{0, 1\}, \quad \forall i \in I, \forall h \in Y_i \end{aligned}$$

In this program, m_d denotes a vector indexed by the terminal joint histories over d (members of Y^d) and containing the expected rewards of these terminal joint histories, l_i denotes $|O_i|^{T-1}$ and D_i denotes the set of subsets in D to which agent i belongs to.

The program is a linearization of Q in that each quadratic term of the objective function of Q is replaced by a linear term (for instance, for $h \in Y_1$ and $\hat{h} \in Y_2$, $x_1(h)x_2(\hat{h})$ is replaced by $z_{12}(h, \hat{h})$). Thus, for each $d \in D$, z_d is a vector of non-integer variables containing one variable per terminal joint history over d . For the 3-chain configuration, the variables of M1 are thus the vectors x_1, x_2, x_3, z_{12} and z_{23} .

The last line of the program ensures that the each x_i is a pure policy. Placing 0-1 constraints on the variables representing terminal histories of each agent is sufficient to ensure that in every solution to the program, even the variables representing nonterminal histories of each agent acquire a value of either 0 or 1.

The constraints of **M1** are explained as follows. Assume we are given a pure joint policy $(\sigma_1, \sigma_2, \dots, \sigma_n)$. Then: (1) The number of terminal histories of agent i that receive a conditional probability of 1 from σ_i is exactly l_i . (2) Therefore, the number of terminal joint histories over d that receive a conditional probability of 1 from the joint policy (σ_i, σ_{-i}) is exactly $l_i l_{-i}$ (where i and $-i$ are used to denote the two agents belonging to d) (3) Moreover, if a terminal history h of agent $i \in d$ receives a conditional probability of 1 from σ_i , the number of terminal joint histories of which h is a part of, and which receive a conditional probability of 1 from (σ_i, σ_{-i}) is exactly l_{-i} .

Note that in the program, we can replace the constraints,

$$l_{-i}x_i(h) = \sum_{J \in Y^d: J_i=h} z_d(J), \quad \forall i \in I, \forall d \in D_i, \forall h \in Y_i$$

by,

$$x_i(J_i) + x_{-i}(J_{-i}) - 2z_d(J) \geq 0, \quad \forall d \in D, \forall J \in Y^d$$

without changing the set of outcomes of the program. However, the constraints of the latter type outnumber by far the constraints of the former type, and hence are not preferable.

Proposition 2. *Given an optimal solution $(x_i^*), \forall i \in I, (z_d^*), \forall d \in D$, to **M1**, $(x_1^*, x_2^*, \dots, x_n^*)$ is an optimal joint policy.*

Proof. The proposition was proved in [1]; the proof is omitted here. \square

We now move to the second 0-1 MIP. Recall that D_i denotes the set of subsets in D to which agent i belongs to. For a terminal history $h \in Y_i$, for a $d \in D_i$ and for a pure policy σ_{-i} define,

$$m_i^d(h, \sigma_{-i}) = \sum_{\hat{h} \in Y_{-i}} r^d(h, \hat{h}) \sigma_{-i}(\hat{h})$$

$-i$ denotes the other agent of the subset d . Furthermore, define,

$$m_i^{d-}(h) = l_{-i} \min_{\hat{h} \in Y_{-i}} r^d(h, \hat{h})$$

$$m_i^{d+}(h) = l_{-i} \max_{\hat{h} \in Y_{-i}} r^d(h, \hat{h})$$

$m_i^{d-}(h)$ and $m_i^{d+}(h)$ are respectively the lower and upper bounds on $m_i^d(h)$ for any σ_{-i} ,

$$m_i^{d-}(h) \leq m_i^d(h, \sigma_{-i}) \leq m_i^{d+}(h)$$

Now consider the following 0-1 MIP.

(M2)

$$\text{maximize} \quad \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i} \{m_i^{d+}(h)x_i(h) + w_i^d(h)\}$$

subject to,

$$x_i \in X_i, \quad \forall i \in I$$

$$w_i^d(h) \leq m_i^d(h, x_{-i}) - m_i^{d+}(h)x_i(h) - m_i^{d-}(h)(1 - x_i(h)),$$

$$\forall i \in I, \forall d \in D_i, \forall h \in Y_i$$

$$w_i^d(h) \leq 0, \quad \forall i \in I \forall d \in D_i, \forall h \in Y_i$$

$$x_i(h) \in \{0, 1\}, \quad \forall i \in I \forall h \in Y_i$$

This program contains one variable $x_i(h)$ for each history h of each agent i . It also contains one variable $w_i^d(h)$ for each history h of each agent i , for each of the subsets d . Notice the absence of variables for *joint* histories in this program. The size of the program is exponential in T but linear in n .

Proposition 3. *Given an optimal solution (x_i^*, w_i^*) , $\forall i \in I$, to **M2**, $(x_1^*, x_2^*, \dots, x_n^*)$ is an optimal joint policy.*

Proof. Note that for each $i = 1$ to n for each $h \in Y_i$ and for each $d \in D_i$,

$$\begin{aligned} w_i^d(h) &= 0, & \text{if } x_i(h) = 0 \\ w_i^d(h) &\leq m_i^d(h, x_{-i}) - m_i^{d+}(h), & \text{if } x_i(h) = 1 \end{aligned}$$

Therefore, we can write,

$$w_i^d(h) \leq \begin{cases} 0, & \text{if } x_i(h) = 0 \\ m_i^d(h, x_{-i}) - m_i^{d+}(h), & \text{if } x_i(h) = 1 \end{cases}$$

This being so, in every *optimal* solution to **M2**, neither of the following two cases arise: (1) $w_i^d(h) < 0$ and $x_i(d) = 0$, (2) $w_i^d(h) < m_i^d(h, x_{-i}) - m_i^{d+}(h)$ and $x_i(h) = 1$ (since we are maximizing, $w_i^d(h)$ will take the largest feasible value instead of the smallest).

Hence we have that,

$$w_i^d(h) = \begin{cases} 0, & \text{if } x_i(h) = 0 \\ m_i^{d+}(h) - m_i^d(h, x_{-i}), & \text{if } x_i(h) = 1 \end{cases}$$

In effect, from the objective function of **M2**, there obtains,

$$\begin{aligned} & \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i} \{m_i^{d+}(h)x_i(h) + w_i^d(h)\} \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i: x_i(h)=1} \{m_i^{d+}(h)x_i(h) + m_i^d(h, x_{-i}) - m_i^{d+}(h)\} \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i: x_i(h)=1} m_i^d(h, x_{-i}) \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i: x_i(h)=1} \sum_{\hat{h} \in Y_{-i}} r^d(h, \hat{h})x_{-i}(\hat{h}) \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} \sum_{h \in Y_i} \sum_{\hat{h} \in Y_{-i}} r^d(h, \hat{h})x_i(h)x_{-i}(\hat{h}) \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{d \in D_i} x_i' M^d x_{-i} \\ &= \sum_{d \in D} x_i' M^d x_{-i} \end{aligned}$$

which is the same objective function as in **Q**. Note that, as stated before, for a $d \in D$ and an $i \in d$, $-i$ denotes the other (than i) agent in d . In other words, in maximizing the objective function of **M2** subject to the constraints on the x_i variables, for $i = 1$ to n , we are effectively maximizing the objective function **Q** subject to the same constraints on the x_i variables. \square

The size of **M1** or **M2** can be reduced by identifying *unrealizable* histories, and not including variables and constraints for such histories in the program. A history of an agent is unrealizable if, given the initial state b^* , the probability that every joint history, of which the history is a part, occurs is 0. Formally, a terminal history h of agent i is unrealizable if,

$$\text{prob.}(h(o), \hat{h}(o)|h(a), \hat{h}(a), b^*) = 0, \quad \forall d \in D_i, \forall \hat{h} \in Y_{-i}$$

Sensor Config.	M1	M2	GOA [4]	SPIDER [6]	SPIDER-ABS [6]
3-chain	1.125	3.73	$\approx 10^3$	$\approx 10^1$	$\approx 10^1$
4-chain	1.148	14.22	$\approx 10^4$	$\approx 10^2$	$\approx 10^2$
5-P	49.3	> 4000	$\approx 10^4$	$\approx 10^4$	$\approx 10^4$
5-star	22.25	3035	$\approx 10^4$	$\approx 10^3$	$\approx 10^3$

Table 1: Time taken in seconds by exact algorithms for solving for horizon 3.

	M^*	Solver	Time Taken (s)	M
3-chain	226	SNOPT	0	120
		LOQO	0.012	163
		LANCELOT	0.26	163
4-chain	338	SNOPT	0.01	70
		LOQO	0.14	248
		LANCELOT	1.31	248

Table 2: Performance of \mathbf{Q} for horizon 3.

Here, as before, $-i$ denotes the agent other than i in set d . $h(o)$ denotes the sequence of observations of h and $h(a)$ denotes the sequence of actions of h . A nonterminal history h of length $t < T$ of an agent is unrealizable if every history of length $t + 1$ of the agent, whose first $2t - 1$ elements coincide with h , is unrealizable. Note that the size of a program can be further reduced by iteratively identifying and excluding *dominated* histories. A dominated history is a history that is provably not required to find an optimal joint policy (an equally good or better history exists). However, this iterated elimination procedure involves solving a series of linear programs, and this in turn can be very time consuming.

6 Computational Experience

We tested the two MIPs, **M1** and **M2**, on the four sensor configurations shown in Figure 1 for horizon 3. The programs were coded in Java and were solved through the branch-and-bound-revised simplex method using ILOG CPLEX. Table 1 shows the time taken by the programs to find an optimal joint policy.

The time taken is inclusive of every computation involved: the calculation of expected rewards of histories, calculation of upper and lower bounds (in **M2**), the identifying of unrealizable histories, the setting up and solving of the program. Also shown in the table is the time taken (approximate, since precise figures were not available) by three existing exact algorithms for this problem.

We also tested \mathbf{Q} on the 3-chain and the 4-chain configurations for horizon 3. \mathbf{Q} was coded in the AMPL language and solved using three freely available QP solvers from the NEOS website¹: SNOPT, LOQO and LANCELOT. The results, given in Table 2, are somewhat discouraging in that while the solvers quickly find a solution, they seem unable to find an optimal joint policy. In the table M^* denotes the expected reward of the optimal joint policy (as found by **M1** and **M2**) while M denotes the expected reward of the locally joint policy found by \mathbf{Q} .

The computational experience is limited to horizon 3. Longer horizons seem out of reach. On the smallest configuration, 3-chain, **M1** took 885 seconds to solve for horizon 4. For the other configurations, in solving for horizon 4, the two programs either cannot be formulated in memory for want of space, or when they can be, take too long to be solved.

7 Discussion

Central to our mathematical programming approach is the use of the sequence form of a policy. In finding an optimal joint policy in the sequence form, we find for each agent a conditional probability distribution over his set of histories. The size of the set of histories of an agent is exponential in T , and

¹<http://neos.mcs.anl.gov/neos/solvers/index.html>

it is reduced substantially (by upto fifty percent in the configurations considered), when unrealizable histories are removed from it. Thus, the use of the sequence form enables us to conceive mathematical programs of a reasonable size (exponential in T : the number of variables and constraints in \mathbf{Q} as well as in $\mathbf{M2}$ is exponential in T and linear in n while it is exponential in $2T$ and linear in n in $\mathbf{M1}$).

As an exact algorithm, $\mathbf{M2}$ is much smaller than $\mathbf{M1}$. However, this advantage in size is not matched by a commensurate advantage in time; indeed, the opposite is seen. $\mathbf{M2}$ takes much longer to be solved than $\mathbf{M1}$. Why does $\mathbf{M2}$ fail where $\mathbf{M1}$ succeeds when both are subject to the same solver (ILOG CPLEX)? The crucial advantage $\mathbf{M1}$ holds over $\mathbf{M2}$ is that the matrix formed by the coefficients of its constraints is *sparse* and the *symmetric*. A working definition of a sparse matrix is that it is a matrix in which the zeros in each row far outnumber the nonzeros in each row. By symmetry, we mean that the zero and nonzero entries in the matrix are arranged in regular patterns. In $\mathbf{M1}$, a typical row consists of a minus one, a very small block of ones and a very large block of zeros. The sparsity and symmetry of $\mathbf{M1}$'s constraints' matrix allows the revised simplex method (used in ILOG CPLEX) to efficiently (rapidly and using little space) solve the relaxation LP of $\mathbf{M1}$ because it reduces the number of arithmetic operations conducted over the tableau and allows a faster inversion of the matrix. When this is not the case, as in $\mathbf{M2}$ whose constraints matrix is neither sparse nor symmetric, ILOG CPLEX falters given the large size of the program. This is one, possibly partial, explanation. A fuller understanding of the problem faced in solving $\mathbf{M2}$ may be arrived at by examining the revised simplex method in detail.

To summarize, we have applied a mathematical programming approach to the problem of multi-target tracking, and have obtained encouraging results when compared to existing approaches. For the configurations of sensors considered, the problem of finding an optimal joint policy reduces to a quadratic program (\mathbf{Q}). We have shown two ways in which \mathbf{Q} can be converted to an exact algorithm. Given the central place occupied by quadratic programming in the domain of nonlinear programming, it may be possible to conceive other ways (other MIPs). Another matter of further investigation could be the conception of approximate algorithms using $\mathbf{M1}$ or $\mathbf{M2}$. As briefly stated before, the size of the MIP can be reduced by identifying all unrealizable or dominated histories. We can thereby use the criteria of ϵ -unrealizability or ϵ -dominance to further whittle down the size of the program in a controlled manner.

References

- [1] R. Aras. Mathematical Programming Methods For Decentralized POMDPs. *Ph.D. Dissertation, Université Henri Poincaré, Nancy*, 2008.
- [2] D. Koller, N. Megiddo, and B. von Stengel. Fast Algorithms For Finding Randomized Strategies In Game Trees. *STOC*, 1994.
- [3] J. Marecki, T. Gupta, P. Varakantham, M. Tambe, and M. Yokoo. Not All Agents Are Equal: Scaling Up Distributed POMDPs For Agent Networks. *AAMAS*, 2008.
- [4] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked Distributed POMDPs: A Synthesis Of Distributed Constraint Optimization And POMDPs. *AAAI*, 2005.
- [5] F. A. Oliehoek, M. T. Spaan, and S. Whiteson. Exploiting Locality Of Interaction In Factored DEC-POMDPs. *AAMAS*, 2008.
- [6] P. Varakantham, J. Marecki, Y. Yabu, M. Tambe, and M. Yokoo. Letting Loose A SPIDER On A Network Of POMDPs: Generating Quality Guaranteed Policies. *AAMAS*, 2007.