A Graphical Modeling Approach to Simplifying Sequential Teams

Aditya Mahajan and Sekhar Tatikonda
Department of Electrical Engineering, Yale University, New Haven, CT -06520, USA
{aditya.mahajan,sekhar.tatikonda}@yale.edu

***Abstract*—A graphical model for sequential teams is presented. This model is easy to understand, and at the same time, is general enough to model any finite horizon sequential team with finite valued system variables and unconstrained decision rules. The model can also be represented as a directed acyclic factor graph. This representation makes it easier to visualize and understand the functional dependencies between different system variables. It also helps in identifying data that is irrelevant for a decision maker to take an optimal decision. Such irrelevant data can be identified using algorithms from graphical models. Thus, the structural properties of optimal decision makers in this model for a sequential team can be identified in an automated manner using the directed acyclic factor graph representation of the sequential team.**

I. Introduction

In this paper we consider decentralized systems consisting of multiple decision makes that have a common objective. A common object implies that the agents cooperate with one another. Due to this cooperation, such systems are also called *decentralized teams*. Within the class of decentralized teams, we restrict attention to systems that operate in discrete time, are synchronous (i.e., all decision makers have a common clock), and are sequential (i.e., the decision makers act in the same order along all sample paths of the underlying random variables). Moreover, we assume that only one decision maker acts at each time and that each decision maker acts once.

Sequential teams are multi-stage decision problems in which the action of a decision maker constitutes a stage. A natural solution concept for multi-stage decision problems is to sequentially decompose the problem into a series of nested optimization subproblems with one subproblem corresponding to one stage. This process is called *sequential decomposition*. The literature on sequential decomposition of sequential teams can be broadly classified into three categories: (i) *General information structures:* Sequential decomposition for arbitrary finite horizon sequential teams is considered in [1]. Sequential decomposition for arbitrary two agent finite and infinite horizon sequential teams is considered in [2]. (ii) *Specific assumptions on information structures:* Sequential decomposition of multi-agent teams under various assumptions on the underlying information structure of the agents has been considered in [3]–[10]. (iii) *Models arising in specific applications:* Sequential decomposition of the following applications have been considered: decentralized Wald problem in [11]; real-time communication in [12]–[16]; networked control system in [17]; multi access channels in [18]. In many instances of a sequential decomposition, a critical first step is to find structural/qualitative properties of optimal decision rules. Normally, finding structural properties requires ingenuity and application specification insight into the problem. In this paper, we present an automated algorithm to derive such structural properties.

There are two types of structural properties: (i) removing redundant data; and (ii) compressing available data using sufficient statistics. Structural properties simplify a sequential team problem in two ways. The first simplification holds for any sequential team problem. The structural properties imply that, without any loss of optimality, we can restrict attention to decision rules that satisfy the structural properties. This restriction reduces the search space for optimal decision rules, thereby, reducing the solution complexity. The second simplification only holds for problems where decision makers correspond to a fixed number of control stations making decisions at different times. In such problems, the structural results may imply that we can restrict attention to decision rules that have a time-invariant domain. This restriction enables extending a sequential decomposition of a finite horizon problem to that of an infinite horizon problem. These simplifications make structural properties desirable. However, finding structural properties is difficult and typically requires ingenuity and application specific insight into the problem.

We focus on the first type structural property and identify conditions under which redundant data can be ignored while taking a decision. In this paper, we present a method to find structural properties of arbitrary sequential teams. This method does not rely on model specific details, rather infers the structural properties from the functional dependence between the system variables. This functional dependence is captured using directed factor graphs, and the structural properties can be derived using standard algorithms from graphical modeling.

Influence diagrams are an existing graphical modeling framework that capture the functional dependencies between system variables in decision problems [19]. However, influence diagram models assume a classical information structure. In contrast, the model presented int his paper allows for a non-classical information structure. Furthermore, influence diagrams determines structural results and sequential decomposition in one go. In sequential teams with non-classical information structure, usually a series of structural results need to be derived before a sequential decomposition can be obtained (see [16], [18] for examples.) For that reason, we believe that it is better to separate the derivation of structural

results from the derivation of a sequential decomposition.

The two main contributions of this paper are: (i) We present a model of sequential teams that is easy to understand, and at the same time, is general enough to model any finite horizon sequential team with finite valued variables and unconstrained decision rules. (ii) We present a algorithm that derives structural results for this model. This gives an automated method to simplify sequential teams.

The rest of this paper is organized as follows. In Section II we present a model for sequential teams and show how such a model can be represented as a directed acyclic factor graph. In Section III we present structural results for the model of Section II and present a algorithm to determine these structural results by restructuring the graph corresponding to the sequential team. We give a few examples in Section IV and conclude in Section V.

II. Modelling and problem formulation

A. Sequential teams

A sequential team consists of the following components.

- A finite set N, two subsets A and R of N, and a partial order $\prec$ on N. The set N indexes system variables, the set A indexes decision makers (DMs), and the set R indexes the rewards.
- A collection $\{\mathcal{X}_n, n \in N\}$ of finite sets representing state spaces of system variables X_n. For any subset S of N, let $X_S \coloneqq (X_n, n \in S)$.
- A collection $\{I_n, n \in N\}$ of subsets of N such that for all $i \in I_n$, $i \prec n$. $\mathcal{I}_n$ denotes $\prod_{i \in I_n} \mathcal{X}_n$.
- A collection $F_{N \setminus A} = \{f_n, n \in N \setminus A\}$ of stochastic kernel kernels, such that f_n is a stochastic kernel from $\mathcal{I}_n$ to $\mathcal{X}_n$, $n \in N \setminus A$. These stochastic kernels represent the system dynamics.

The design objective is to choose decision functions $G_A \coloneqq (g_n, n \in A)$, where g_n is a function from $\mathcal{I}_n$ to $\mathcal{X}_n$, $n \in A$, so as to maximize $\mathbb{E}\left\{\sum_{n \in R} X_n\right\}$. The expectation is with respect to a joint measure P^{G_A} on X_N which is given by

$$P^{G_A}(X_N) = \prod_{n \in N \setminus A} f_n(X_n | I_n) \prod_{n \in A} \mathbb{1}\left[X_n = g_n(I_n)\right]. \qquad (1)$$

B. Some remarks on the model

The sequential team model of [1] with finite system variable and finite primitive random variables can be reduced to the sequential team model defined above. The model of [1] assumes a total order on all agents. When the system variables and primitive random variables are finite valued, the σ-algebras $\mathcal{J}_t$ defined in [1] are finite. So, these σ-algebras can be generated by finite partitions. The atoms of these partitions can be considered as the data observed by the DMs. This gives rise to a sequential team model of the type defined above.

The sequential form of the intrinsic model defined in [20] when finite system and finite primitive random variables can be reduced to the sequential model defined above. The intrinsic model is sequential if and only if there is a partial order on all system variables. When the system variables and primitive random variables are finite valued, the σ-algebras $\mathcal{J}_\alpha$ defined in [20] are finite. So, as in the model of [1], these σ-algebras can be generated by a finite partition and the atom of these partitions can be considered as the data observed by the DMs. This gives rise to a sequential team model of the type defined above.

The sequential team problem defined above is an unconstrained optimization problem where the decision rules $\{g_n, n \in A\}$ can be any function from $\mathcal{I}_n$ to $\mathcal{X}_n$. Hence, using randomized decision rules will not improve performance [20]. So there is no loss of generality in assuming that all decision rules are deterministic.

C. A graphical model for decentralized teams

The sequential team defined above can also be modeled as a labelled directed acyclic factor graph (DAFG). A DAFG is a bipartite graph with two types of nodes, variable nodes and factor nodes, that are connected by directed edges. Formally, a DAFG $\mathcal{G}$ is a tuple $(\mathcal{V}, \mathcal{F}, \mathcal{E})$, where $\mathcal{V}$ is the set of variable nodes, $\mathcal{F}$ is the set of factor nodes, and $\mathcal{E}$ is the set of directed edges, which is a subset of $(\mathcal{V} \times \mathcal{F}) \cup (\mathcal{F} \times \mathcal{V})$. Each node has a label $\{0, 1\}$; for variable node, a label 1 indicates that it is a reward node; for factor nodes, a label 1 indicates that it is a decision rule factor. Moreover, the graph is acyclic, so $\mathcal{E}$ has no directed cycles.

A sequential team can be represented as a DAFG $\mathcal{G}(\mathcal{V}, \mathcal{F}, \mathcal{E})$. The variable nodes $\mathcal{V}$ and the factor nodes $\mathcal{F}$ are both indexed by N. To avoid confusion, we will use a tilde above the number to indicate the index of a factor node. The variable nodes with index in R have a label 1, others have a label 0; the factor nodes with index in A have a label 1, others have a label 0. The set of edges $\mathcal{E}$ is given by $\{(i, \tilde{n}) : i \in I_n, n \in N\} \cup \{(i, \tilde{n}) : i \in I_n, n \in A\} \cup \{(\tilde{n}, n) : n \in N\}$ Thus, there is a variable node corresponding to each system variable and a factor node corresponding to each stochastic kernel and decision rules. These nodes are connected to indicate the functional dependencies between them. Furthermore, there is only one incoming edge at each variable node, and only one outgoing edge from each factor node. When we draw a DAFG, we use circles to indicate variable nodes and squares to indicate factor nodes. We indicate reward variables by solid circles and decision rule factors by solid squares.

We will use some terminology from graphical models. If there is an edge between node a and node b, then a is said to be *parent* of b and b a *child* of a, and a and b are *neighbors*. The set of parents of b is denoted as $\mathrm{pa}(b)$, the set of children of a as $\mathrm{ch}(a)$, the set of neighbors of a as $\mathrm{ne}(a)$. There is a *path* from a to b if there exists nodes $a = a_0, a_1, \ldots, a_k = b$ such that $(a_{i-1}, a_i) \in \mathcal{E}$, $i = 1, \ldots, k$. There is a *trail* from a to b if there exists nodes $a = a_0, a_1, \ldots, a_k = b$ such that either $(a_{i-1}, a_i) \in \mathcal{E}$ or $(a_i, a_{i-1}) \in \mathcal{E}$, $i = 1, \ldots, k$. The nodes a such that there is a path from a to b are called *ancestors* $\mathrm{an}(b)$ of b. The nodes b such that there is a path from a to b are called *descendants* $\mathrm{de}(a)$ of a.

D. An example

To fix ideas, we consider the simplest sequential team, namely a Markov decision process (MDP), and show how to model it by a DAFG. A MDP consists of a plant and a control station. The control station observes the state of the plant and takes a control action based on all past plant observations and all past control actions. At each time an instantaneous reward that depends on the current state of the plant and the current control action is received. The control objective is to choose control laws to maximize the total expected reward.

We consider a MDP that operates for three time steps. First, we describe a mathematical model for the MDP; then, we construct a DAFG corresponding to that model.

Let x_1, x_2, x_3 denote the state of the plant, u_1, u_2, u_3 denote the control actions, and r_1, r_2, r_3 denote the rewards. A partial order $\prec$ on these variables is given by $x_1 \prec u_1 \prec x_2 \prec u_2 \prec x_3 \prec u_3$, and $x_i \prec r_i$, $u_i \prec r_i$, $i = 1, 2, 3$.

The plant dynamics are given by stochastic kernels $f_0(x_1)$, $f_1(x_2|x_1, u_1)$, and $f_2(x_3|x_2, u_2)$. The rewards are given by $\rho_1(r_1|x_1, u_1)$, $\rho_2(r_2|x_2, u_2)$, and $\rho_3(r_3|x_3, u_3)$. The control station acts at three time instances. Therefore, the system has three decision makers, DM 1, DM 2, and DM 3. DM i generates control action u_i according to the following rules.

$$\begin{aligned} u_1 &= g_1(x_1), \\ u_2 &= g_2(x_1, x_2, u_1), \\ u_3 &= g_3(x_1, x_2, x_3, u_1, u_2). \end{aligned}$$

Now, we construct a DAFG corresponding to the above MDP model. For ease of representation, we will index the variable and factor nodes by the name of the variable. The DAFG has 9 variable nodes: x_1, x_2, x_3, u_1, u_2, u_3, r_1, r_2, r_3, and 9 factor nodes: f_0, f_1, f_2, g_1, g_2, g_3, ρ_1, ρ_2, ρ_3. The variable nodes r_1, r_2, r_3 are reward nodes, and the factor nodes g_1, g_2, g_3 are decision rule factors. Directed edges are added between the nodes to indicate functional dependencies between the nodes. The resulting DAFG is shown in Fig. 1.

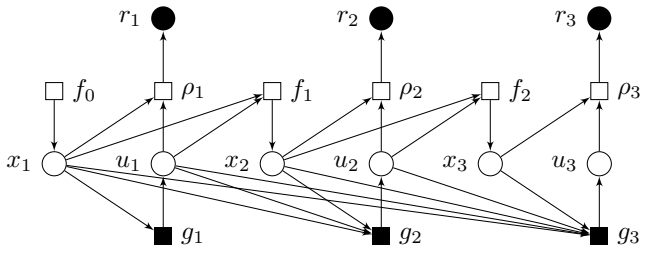


Fig. 1: A DAFG corresponding to a MDP that operates for three time steps.

The design objective is to choose decision rules G_A, which are shown by solid squares, to maximize the expected value of the sum of the reward variables, which are shown by solid circles.

In this paper, we simplify the design of sequential teams by transforming the corresponding DAFG. These graph transformations are based on properties of measures that recursively factorize with respect to a DAFG. In the next section, we explain these measures and their properties.

E. Recursive factorization of probability measures

Given a DAFG corresponding to a sequential team, a joint measure on X_N is said to *recursively factorize* according to the DAFG $\mathcal{G}$ if there exists stochastic kernels $\hat{F}_{N\setminus A} = \{\hat{f}_n, n \in N \setminus A\}$ and functions $\hat{G}_A = \{\hat{g}_n, n \in A\}$ such that

$$Q(X_N) = \prod_{n \in N\setminus A} \hat{f}_n(\mathrm{ch}(\tilde{n})|\,\mathrm{pa}(\tilde{n})) \prod_{n \in A} \mathbb{1}\left[\mathrm{ch}(\tilde{n}) = \hat{g}_n(\mathrm{pa}(\tilde{n}))\right] \tag{2}$$

where the equality holds Q almost surely.

The joint measure P^{G_A} on X_N, which is given by (1), is of the form (2). So, any choice of decision rules G_A induces a joint measure P^{G_A} on X_N that recursively factorizes according to the DAFG $\mathcal{G}$.

We now define some properties of graphical models which are invariant for any measure Q that recursively factorizes according to $\mathcal{G}$. The variables X_J are said to be *irrelevant* to X_L given X_K in $\mathcal{G}$, denoted by $X_J \perp\!\!\!\perp_{\mathcal{G}} X_L \mid X_K$, if for any probability measure Q that recursively factorizes according to $\mathcal{G}$

$$Q(X_J|X_K, X_L) = Q(X_J|X_K)$$

The set of all variables that are irrelevant to X_J given X_K are called irrelevant variables and denoted by $\mathrm{ir}(X_J|X_K)$. The set of variable nodes that are irrelevant to X_N given X_K are called nodes *functionally determined* by X_K and denoted by $\mathrm{fd}(X_K)$. Thus, $\mathrm{fd}(X_K) = \mathrm{ir}(X_N|X_K)$.

For any probability Q that recursively factorizes according to $\mathcal{G}$, the conditional probability $Q(X_J|X_K)$ can be expressed in terms of the marginals of (2). In this ratio of marginals, some product terms are common that will get cancelled. The remaining terms indicate the factor nodes that are needed to compute the conditional probability $Q(X_J|X_K)$. These factor nodes are called *requisite factor nodes* and denoted by $\mathrm{re}(X_J|X_K)$. Furthermore, the subset $\mathrm{ob}(X_J|X_K)$ of X_K whose values are needed to compute the conditional probability $Q(X_J|X_K)$ are called *observation variables*. The notion of requisite factor nodes is similar to that of requisite probability nodes defined in [22]. The only difference is that we explicitly keep track of the factors.

The *effective observations* for X_J given X_K, denoted by $\mathrm{ef}(X_J|X_K)$, are the subset of nodes functionally determined by X_K whose parents are requisite factors for X_J given X_K. Formally,

$$\mathrm{ef}(X_J|X_K) = \{n \in \mathrm{fd}(X_K) : \mathrm{pa}(n) \in \mathrm{re}(X_J \,|\, X_K)\} \setminus X_K$$

All the sets defined for X_J given X_K defined above, $\mathrm{ir}(X_J|X_K)$, $\mathrm{re}(X_J|X_K)$, $\mathrm{ob}(X_J|X_K)$, and $\mathrm{ef}(X_J|X_K)$, can be determined using D-separation [23], which is an extension of d-separation [24] when there are deterministic functions. D-separation in DAFG can be checked in linear time, either using a variation of breadth first search [23] or using a variation of the Bayes Ball algorithm [22]. Therefore, the sets $\mathrm{ir}(X_J|X_K)$, $\mathrm{re}(X_J|X_K)$, $\mathrm{ob}(X_J|X_K)$, and $\mathrm{ef}(X_J|X_K)$ can be determined in linear time.

F. Total order on decision makers

In the result presented in this paper, we need a total order on all decision makers. In general, there are many total orders compatible with the partial order $\prec$. Given any partial order, a total order compatible with that partial order can be obtained using topological sort [21], which has a complexity linear in the number of nodes and edges.

III. Structural result

All the data available at a decision maker may not be necessary to take an optimal decision. In this section, we find conditions to identify the redundant data at a decision maker. We also find conditions under which effectively observed data can be used by the decision maker. Then, we present a graphical test to check these conditions, and a graph transform to prune the redundant data and add effectively observed data at each DM.

A. Statement of the structural result

Structural results for DM n, $n \in A$, can be derived by fixing the decision rules for all other DMs. With respect to DM n, the reward variables can be partitioned into two sets, dependent reward variables $R_d(\tilde{n})$ and independent reward variables $R_i(\tilde{n})$. These sets are defined as

$$R_d(\tilde{n}) \coloneqq \mathrm{de}(\tilde{n}) \cap R, \quad R_i(\tilde{n}) \coloneqq R \setminus R_d(\tilde{n}).$$

The main result of this paper is the following.

Theorem 1: In the sequential team model of Section II-A, the decision rules are of the form

$$u_n = g_n(\mathrm{pa}(\tilde{n})).$$

However, there is no loss of optimality in choosing rules of the form

$$u_n = g_n\big(\mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})) \setminus \{u_n\}, \mathrm{ef}(R_d(\tilde{n})|\,\mathrm{pa}(\tilde{n})) \setminus \mathrm{de}(\tilde{n})\big) \quad (3)$$

Proof: The proof is along the lines of the proof of the three step lemma of [25]. We fix the decision rules all DMs other than DM n, and look at the problem from the point of view of DM n. Since for any $r \in R_i(\tilde{n})$, $\tilde{n} \not\prec r$, the choice of g_n does not influence the values of $R_i(\tilde{n})$. Thus,

$$\begin{aligned}
&\max_{g_n} \mathbb{E}^{G_A, F_{N\setminus A}} \left\{ \sum_{X_i \in R} X_i \,\middle|\, \mathrm{pa}(\tilde{n}) \right\} \\
&\quad = \mathbb{E}^{G_{A\setminus(\mathrm{de}(\tilde{n})\cup\{\tilde{n}\})}, F_{N\setminus(A\cup\mathrm{de}(\tilde{n}))}} \left\{ \sum_{X_i \in R_i(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}) \right\} \\
&\qquad + \max_{g_n} \mathbb{E}^{G_A, F_{N\setminus A}} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}) \right\}
\end{aligned}$$

Now, the second term can be simplified as follows.

$$\begin{aligned}
&\mathbb{E}^{G_A, F_{N\setminus A}} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}) \right\} \\
&\overset{(a)}{=} \mathbb{E}^{G_A, F_{N\setminus A}} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}), u_n \right\} \\
&\overset{(b)}{=} \mathbb{E}^{\mathrm{re}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n}))} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}), u_n \right\} \\
&\overset{(c)}{=} \mathbb{E}^{\mathrm{re}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n}))\setminus\{g_n\}} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{pa}(\tilde{n}), u_n \right\} \\
&\overset{(d)}{=} \mathbb{E}^{\gamma} \left\{ \sum_{X_i \in R_d(\tilde{n})} X_i \,\middle|\, \mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})), u_n \right\} \\
&\overset{(e)}{=:} F_\gamma(\mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})) \setminus \{u_n\}, u_n)
\end{aligned}$$

In the above expression, (a) follows from $u_n = g_n(\mathrm{pa}(\tilde{n}))$; (b) follows from the definition of requisite factors; (c) follows from policy independence of conditional expectation [26]; (d) follows from definition of observation variables; the term γ is (d) is a short hand for $\mathrm{re}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})) \setminus \{g_n\}$; and the function F_γ in (e) is for notational convenience.

Now, let $U = u_n$, $Y = \mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})) \setminus \{u_n\}$, and $Z = \mathrm{pa}(\tilde{n}) \setminus Y$. Then, the optimization problem at DM n is

$$\max_{g_n} F_\gamma(Y, g_n(Y, Z)).$$

Now, let $\hat{g}(Y) = \arg\max_U F_\gamma(Y, U)$ where ties are broken according to arbitrary but fixed rule. Then,

$$F(Y, \hat{g}(Y)) \geq F_\gamma(Y, U) = F(Y, g_n(Y, Z)) \quad (4)$$

for all U, Y, and Z. When the decision space Y belongs to an infinite space, we need to worry about measurability of $\hat{g}$. In that case, a measurable $\hat{g}$ that satisfies (4) can be chosen by breaking ties in a specific manner. For example, see [25], [27].

Equation (4) implies that we can choose an optimal decision rule $\hat{g}$ for DM n such that

$$u_n = \hat{g}(\mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n})) \setminus \{u_n\}).$$

The choice of $\hat{g}$ depends on γ. Furthermore, the variables $\mathrm{ef}(R_d(\tilde{n})|\,\mathrm{pa}(\tilde{n})) \setminus \mathrm{de}(\tilde{n})$ depend only on γ and do not depend on u_n or g_n. Since, all functions in γ are fixed, the DM n can compute $\mathrm{ef}(R_d(\tilde{n})|\,\mathrm{pa}(\tilde{n})) \setminus \mathrm{de}(\tilde{n})$ and use them for taking a decision. Thus, without loss of optimality all DMs can restrict attention to decision rules of the form (3). ∎

B. Graphical algorithm for structural results

We need to identify $\mathrm{de}(\tilde{n})$, $\mathrm{pa}(\tilde{n})$, $R_d(\tilde{n})$, $\mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n}))$ and $\mathrm{ef}(R_d(\tilde{n})|\,\mathrm{pa}(\tilde{n}))$ to reduce the decision rule at DM n to the form (3). The sets $\mathrm{de}(\tilde{n})$, $\mathrm{pa}(\tilde{n})$ and $R_d(\tilde{n})$ can be computed efficiently from the graphical model. The sets $\mathrm{ob}(R_d(\tilde{n})|\,\mathrm{ne}(\tilde{n}))$ and $\mathrm{ef}(R_d(\tilde{n})|\,\mathrm{pa}(\tilde{n}))$ can be computed

in linear time [22], [23]. Thus, we can efficiently find the structural results of all DMs using a graphical algorithm, which we call graph restructuring. This algorithm is shown in Algorithm 1. We pick DMs in an descending order according to $\prec$, and apply the structural results of Theorem 1 to each DM one by one. The graph restructuring algorithm can be applied multiple times until it does not change the DAFG. This algorithm is called graph simplification and shown in Algorithm 2.

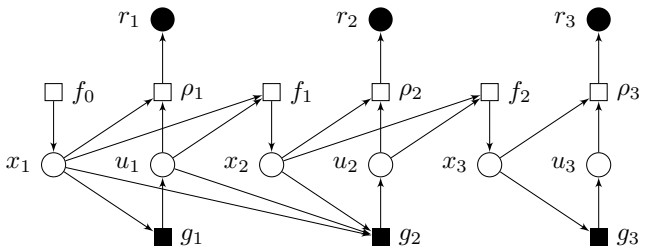
Algorithm 1 Graph restructuring algorithm

function RESTRUCTURE_GRAPH ($\mathcal{G}(\mathcal{V}, \mathcal{F}, \mathcal{E})$, A, R)
  $A_s \leftarrow$ topological_sort(A)
  $A_r \leftarrow$ reverse(A_s)
  for all $\tilde{n} \in A_r$ **do**
    $Pa \leftarrow \mathrm{pa}(\tilde{n})$
    $U \leftarrow \{u_n\}$
    $Ne \leftarrow Pa \cup U$
    $De \leftarrow \mathrm{de}(\tilde{n})$
    $Rd \leftarrow De \cap R$
    $Ob \leftarrow \mathrm{ob}(Rd|Ne) \setminus U$
    $Ef \leftarrow \mathrm{ef}(Rd|Pa) \setminus De$
    for all $a \in Pa \setminus Ob$ **do**
      $\mathcal{E} \leftarrow \mathcal{E} \setminus \{(a, \tilde{n})\}$
    end for
    for all $a \in Ef \setminus De$ **do**
      $\mathcal{E} \leftarrow \mathcal{E} \cup \{(a, \tilde{n})\}$
    end for
    $\mathcal{G} \leftarrow \mathcal{G}(\mathcal{V}, \mathcal{F}, \mathcal{E})$
  end for
  return $\mathcal{G}$
end function

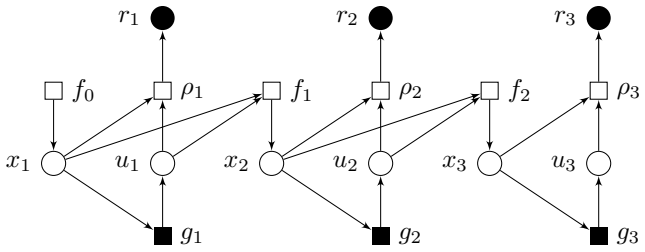
Algorithm 2 Graph simplification algorithm

function SIMPLIFY_GRAPH($\mathcal{G}$, A, R, max_count)
  count $\leftarrow 0$
  repeat
    count $\leftarrow$ count + 1
    $\mathcal{G}' \leftarrow \mathcal{G}$
    $\mathcal{G} \leftarrow$ restructure_graph($\mathcal{G}$, A, R)
  until ($\mathcal{G} = \mathcal{G}'$) or (count = max_count)
  return $\mathcal{G}$
end function

To understand this algorithm, lets consider the MDP of Section II-D. After sorting the DMs, we get g_3, g_2, g_1. For DM g_3, $\mathrm{pa}(g_3) = \{x_1, u_1, x_2, u_2, x_3\}$, $R_d(g_3) = \{r_3\}$, $\mathrm{ob}(R_d(g_3)|\,\mathrm{ne}(g_3)) = \{x_3, u_3\}$, and $\mathrm{ef}(R_d(g_3)|\,\mathrm{pa}(g_3)) = \emptyset$. Thus, the edges (x_1, g_3), (u_1, g_3), (x_2, g_3), and (u_2, g_3) are removed and no edges are added. The resultant DAFG is shown in Fig. 2(a). For DM g_2 in this DAFG, $\mathrm{pa}(g_2) = \{x_1, u_1, x_2\}$, $R_d(g_2) = \{r_2, r_3\}$, $\mathrm{ob}(R_d(g_2)|\,\mathrm{ne}(g_2)) = \{x_2, u_2\}$, and $\mathrm{ef}(R_d(g_2)|\,\mathrm{pa}(g_2)) = \emptyset$. Thus, the edges (x_1, g_2) and (u_1, g_2) are removed. The resultant DAFG is shown in Fig. 2(b). For DM g_1 in this DAFG, $\mathrm{pa}(g_1) = \{x_1\}$, $R_d(g_1) = \{r_1, r_2, r_3\}$, $\mathrm{ob}(R_d(g_1)|\,\mathrm{ne}(g_1)) = \{x_1, u_1\}$, and $\mathrm{ef}(R_d(g_1)|\,\mathrm{ne}(g_1)) = \emptyset$. Thus, no edge is removed or added. Thus, DAFG of Fig. 1 can be reduced to DAFG of Fig. 2(b) without any loss of optimality.



(a) Graph restructuring to node g_3 of MDP of Fig 1.



(b) Graph restructuring to node g_2 of MDP of part (a).

Fig. 2: Intermediate steps of graph restructuring algorithm to MDP shown in Fig. 1.

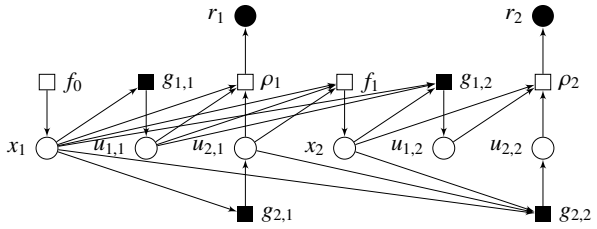
The first step of Algorithm 1, which reverse sorts the DMs according to the partial order $\prec$, is optional. If we pick DMs in an arbitrary order, the graph simplification algorithm will take more steps before it converges. In Algorithm 2, we believe that after a finite number of steps $\mathcal{G}'$ should be equal to $\mathcal{G}$. However, we have not been able to prove this so far. Thus, to ensure the correctness of the algorithm, we limit the number of iterations to max_count.

IV. Some illustrative examples

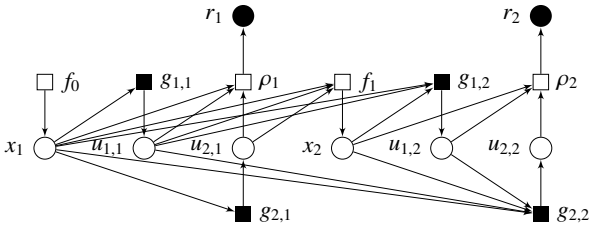
A. A two agent sequential team

Consider a sequential team with two control stations that runs for 2 time steps. First, we describe a mathematical model for this sequential team; next, we construct a DAFG corresponding to that model; then, we apply the graph simplification algorithm on this DAFG.

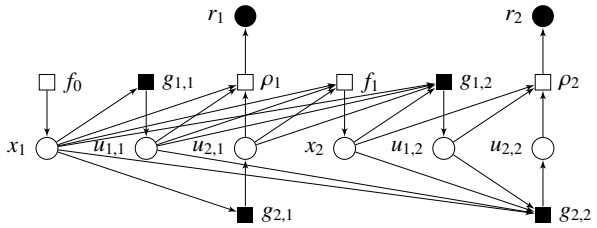
Let x_1, x_2 denote the state of the plant, $u_{1,1}$, $u_{1,2}$ denote the control actions of station 1, $u_{2,1}$, $u_{2,2}$ denote the control actions of station 2, and r_1, r_2, denote the rewards. The plant dynamics are given by stochastic kernels $f_0(x_1)$, $f_1(x_2|x_1, u_{1,1}, u_{2,1})$. The rewards are given by stochastic kernels $\rho_1(r_1|x_1, u_{1,1}, u_{2,1})$, $\rho_2(r_2|x_2, u_{1,2}, u_{2,2})$. The two control stations act for two time instances. Therefore, the system has four decision makers, DM (1,1), DM (1,2), DM (2,1), and DM (2,2). DM (i, j) generates control action $u_{i,j}$ according



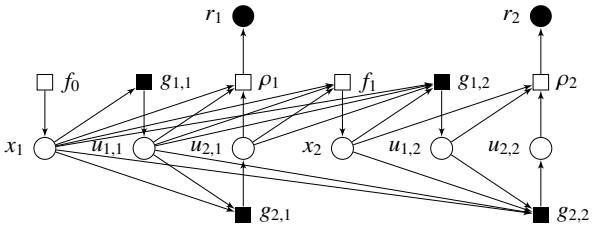
(a) DAFG corresponding to a simple two agent team.



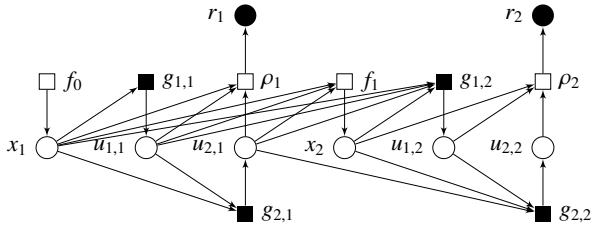
(b) First pass of graph simplification. Graph restructuring at factor $g_{2,2}$ of the DAFG of part (a). Edge $(u_{2,1}, g_{2,2})$ is removed and edges $(u_{1,1}, g_{2,2})$ and $(u_{1,2}, g_{2,2})$ are added.



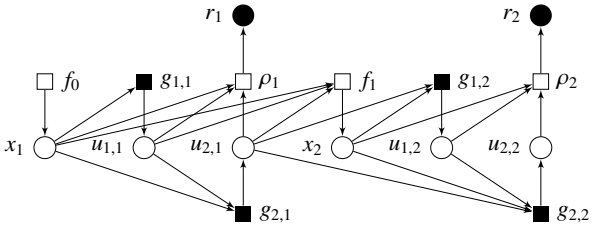
(c) First pass of graph simplification. Graph restructuring at factor $g_{2,1}$ of the DAFG of part (b). Edge $(u_{2,1}, g_{1,2})$ is added.



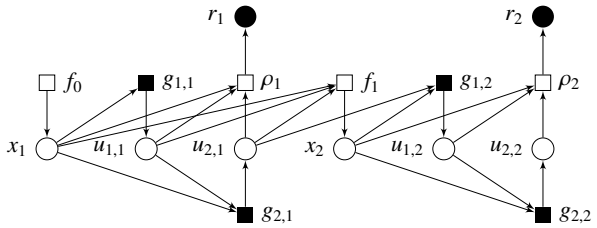
(d) First pass of graph simplification. Graph restructuring at factor $g_{1,2}$ of the DAFG of part (c). Edge $(u_{1,1}, g_{2,1})$ is added.



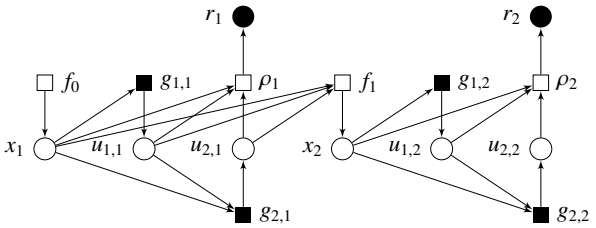
(e) Second pass of graph simplification. Graph restructuring at factor $g_{2,2}$ of the DAFG of part (d). Edge $(u_{2,1}, g_{2,2})$ is added and edges $(x_1, g_{2,2})$ and $(u_{1,1}, g_{2,2})$ are removed.



(f) Second pass of graph simplification. Graph restructuring at factor $g_{1,2}$ of the DAFG of part (e). Edges $(x_1, g_{1,2})$ and $(u_{1,1}, g_{1,2})$ are removed.



(g) Third pass of graph simplification. Graph restructuring at factor $g_{2,2}$ of the DAFG of part (f). Edge $(u_{2,1}, g_{2,2})$ is removed.



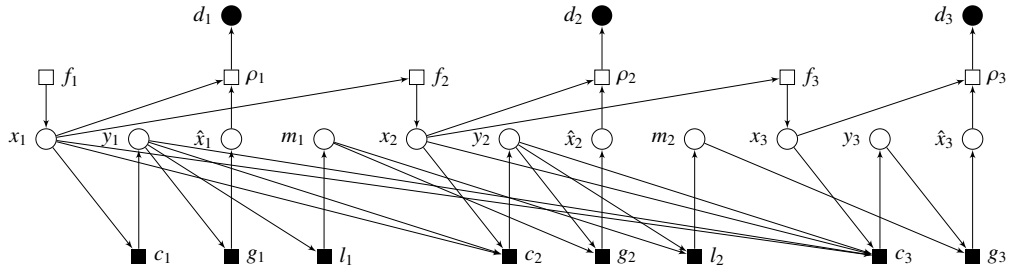
(h) Third pass of graph simplification. Graph restructuring at factor $g_{1,2}$ of the DAFG of part (g). Edge $(u_{2,1}, g_{1,2})$ is removed.

Fig. 3: A DAFG corresponding to a simple two station sequential team and its simplification.

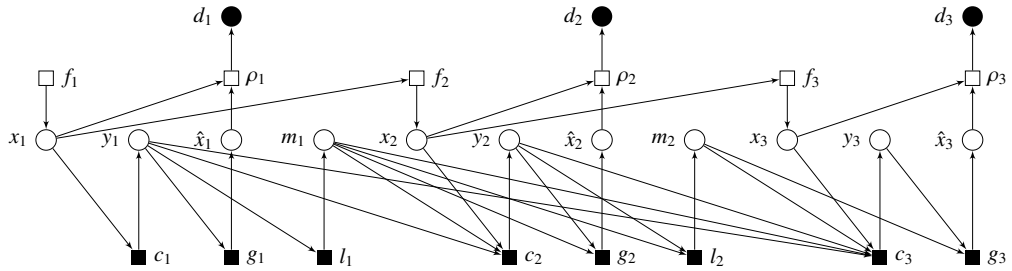
to the following rules.

$$u_{1,1} = g_{1,1}(x_1), \qquad u_{2,1} = g_{2,1}(x_1),$$
$$u_{1,2} = g_{1,1}(x_1, x_2, u_{1,1}), \qquad u_{2,2} = g_{2,1}(x_1, x_2, u_{2,1}).$$

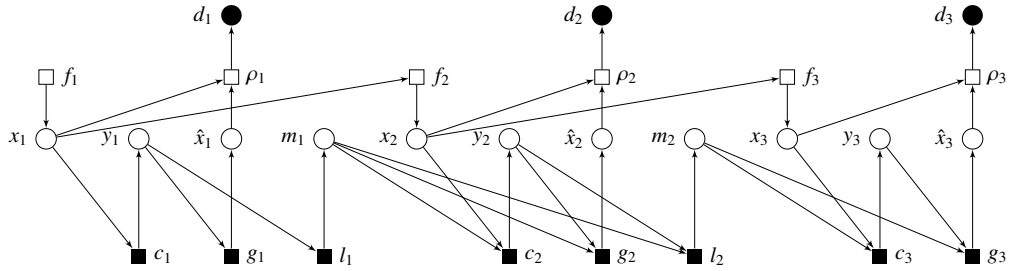
A DAFG corresponding to this model is shown in Fig. 3(a). For ease of notation, we represent the graph nodes by names of the variables and the functions, rather than an index. For this DAFG, a total order on the DMs is $\{g_{1,1}, g_{2,1}, g_{1,2}, g_{2,2}\}$. The simplification of this model, including the intermediate steps, is shown in Fig. 3(b)–(h). The graph simplication requires three passes before settling on a result. A third pass could have been avoided if we had kept track of nodes which were deteled in the previous passes. For example, while running graph restructuring at DM $g_{2,2}$ the edge $(u_{2,1}, g_{2,2})$ was deleted in the first pass, added back in the second pass, and deleted again in the third pass. Had we not adding this edge back in the second pass, the third pass would have been avoided. This suggests a possible fine tuning of graph restructuring algorithm



(a) DAFG corresponding to a real-time source coding system.



(b) Graph simplification of the DAFG of part (a). Edges (m_1, c_3), (m_2, c_3), (m_1, c_2) are added and edges (x_1, c_3), (x_2, c_3), (x_1, c_2) are deleted.



(c) Graph simplification of the DAFG of part (b). Edges (y_1, c_3), (y_2, c_3), (m_1, c_3), (y_1, c_2) are removed.

Fig. 4: Graph simplification of a DAFG of a real-time source coding system.

at the cost of keeping track of the results of previous passes.

Fig. 3(h) shows that there is no loss optimality in choosing control laws of the form

$$u_{1,1} = g_{1,1}(x_1), \qquad u_{2,1} = g_{2,1}(x_1, u_{1,1}),$$
$$u_{1,2} = g_{1,1}(x_2), \qquad u_{2,2} = g_{2,1}(x_2, u_{1,2}).$$

This structural result can also be derived analytically by following the same reasoning as the graph simplification.

B. Real time source coding

Consider the real-time source coding problem described in [25] that operates for 3 time steps. The system variables are given as follows. Let x_1, x_2, x_3, denote the outputs of a Markov source. These are encoded casually and in real-time by an encoder to produce encoded symbols y_1, y_2, y_3. These encoded symbols are observed by a finite memory receiver. The receiver updates its memory m_1, m_2, and generates source estimates $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$ causally and in real-time. A distortion d_1, d_2, d_3, is incurred at each step. A partial order on these system variables is given by $x_1 \prec x_2 \prec x_3$, $y_1 \prec y_2 \prec y_3$, $m_1 \prec m_2$, $x_i \prec y_i \prec \hat{x}_i \prec d_i$, $i = 1, 2, 3$, and $y_i \prec m_i \prec \hat{x}_{i+1}$, $i = 1, 2$.

The system stochastic kernels and decision rules are given as follows. The source output is given by stochastic kernels $f_1(x_1)$, $f_2(x_2|x_1)$, $f_3(x_3|x_2)$. The distortions are given by $\rho_1(d_1|x_1, \hat{x}_1)$, $\rho_2(d_2|x_2, \hat{x}_2)$, $\rho_3(d_3|x_3, \hat{x}_3)$. The decision rules at the encoder and receiver are given by

$$y_1 = c_1(x_1), \qquad \hat{x}_1 = g_1(y_1)$$
$$y_2 = c_2(x_1, x_2, y_1) \qquad \hat{x}_2 = g_2(y_2, m_1)$$
$$y_3 = c_2(x_1, x_2, x_3, y_1, y_2) \qquad \hat{x}_3 = g_3(y_3, m_2)$$
$$m_1 = l_1(y_1) \qquad m_2 = l_2(y_2, m_1)$$

The objective is to choose c_1, c_2, c_3, g_1, g_2, g_3, l_1, and l_2 to minimize the total expected distortion (which is the same as maximizing the negative of total expected distortion).

A DAFG corresponding to the above model is shown in Fig. 4(a). For ease of notation, we represent the graph nodes by the names of the variables and functions, rather than an index. For this DAFG, a total or-

der on the DMs is $\{c_1, g_1, l_1, c_2, g_2, l_2, c_3, g_3\}$. For DM g_3, $\mathrm{ob}(R_d(g_3)|\,\mathrm{ne}(g_3)) = \mathrm{ne}(g_3)$ and $\mathrm{ef}(R_d(g_3)|\,\mathrm{pa}(g_3)) = \emptyset$. Thus, we do not add or remove any edges. For DM c_3, $\mathrm{ob}(R_d(c_3)|\,\mathrm{ne}(c_3)) = \{x_3, y_1, y_2, y_3, u_3\}$ and $\mathrm{ef}(R_d(c_3)|\,\mathrm{pa}(c_3)) = \{m_1, m_2\}$. Thus, we remove edges (x_1, c_3) and (x_2, c_3) and add edges (m_1, c_3) and (m_2, c_3). Continuing this way, for DM l_2 and g_2, we neither add nor remove any edges; for DM c_2, edge (x_1, c_1) is removed and edge (m_1, c_1) is added. For DMs l_1, g_1, and c_1, no edges are added or removed. The resultant DAFG is shown in Fig. 4(b). We once again apply the graph simplification algorithm to the DAFG of Fig. 4(b). In this second pass, the edges (y_1, c_3), (y_2, c_3), (m_1, c_3), (y_1, c_2) are removed. The resultant DAFG is shown in Fig. 4(c), which shows that there is no loss of optimality in restricting attention to the encoding rules of the form

$$y_1 = c_1(x_1), \quad y_2 = c_2(x_2, m_1), \quad y_3 = c_3(c_3, m_2).$$

This is the same as the structural results proved analytically in [25].

Notice that both these example required multiple passes to settle on a structural result. This supports our proposal of separating structural results from sequential decomposition. The second and higher rounds of structural results will be missed by algorithms like those in influence diagrams that performs strucutral results and sequential decomposition in a single step.

V. Conclusion

We present a model for sequential teams that can be represented as a DAFG. This model can be used as a pedagogical tool for understanding structural results for sequential teams as well as a computational tool for deriving such structural results in an automated manner. Broadly, there are two types of structural results in sequential teams, namely, removing redundant data and compressing available data. In this paper, we present an algorithm based on graphical models that can indentify redundant data available at a decision maker. We believe that an algorithm for compressing available data a decision maker can also be identified.

Acknowledgements

The first author is grateful to Demosthenis Teneketzis, Ashutosh Nayyar, and Achilleas Anastasopoulos for extensive discussions that crystalized the ideas presented in this paper. We also thank Achilleas Anastasopoulos and Ashutosh Nayyar for detailed feedback on an earlier draft of this paper and for extensive testing of a software implementation of the algorithm presented in this paper.

References

[1] H. S. Witsenhausen, "A standard form for sequential stochastic control," *Mathematical Systems Theory*, vol. 7, no. 1, pp. 5–11, 1973.

[2] A. Mahajan, "Sequential decomposition of sequential dynamic teams: applications to real-time communication and networked control systems," Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 2008.

[3] Y.-C. Ho and K.-C. Chu, "Team decision theory and information structures in optimal control problems–Part I," *IEEE Trans. Autom. Control*, vol. 17, no. 1, pp. 15–22, 1972.

[4] P. Varaiya and J. Walrand, "On delayed sharing patterns," *IEEE Trans. Autom. Control*, vol. 23, no. 3, pp. 443–445, 1978.

[5] T. Yoshikawa, "Decomposition of dynamic team decision problems," *IEEE Trans. Autom. Control*, vol. 23, no. 4, pp. 627–632, Aug. 1978.

[6] G. Casalino, F. Davoli, R. Minciardi, P. Puliafito, and R. Zoppoli, "Partially nested information structures with a common past," *IEEE Trans. Autom. Control*, vol. 29, no. 9, pp. 846–850, Sep. 1984.

[7] M. Aicardi, F. Davoli, and R. Minciardi, "Decentralized optimal control of Markov chains with a common past information set," *IEEE Trans. Autom. Control*, vol. 32, no. 11, pp. 1028–1031, 1987.

[8] J. M. Ooi, S. M. Verbout, J. T. Ludwig, and G. W. Wornell, "A separation theorem for periodic sharing information patterns in decentralized control," *IEEE Trans. Autom. Control*, vol. 42, no. 11, pp. 1546–1550, Nov. 1997.

[9] A. Mahajan, A. Nayyar, and D. Teneketzis, "Identifying tractable decentralized control problems on the basis of information structures," in *proceedings of the 46th Allerton conference on communication, control and computation*, Sep. 2008, pp. 1440–1449.

[10] S. Yüksel, "Stochastic nestedness and the belief sharing information pattern," *IEEE Trans. Autom. Control*, 2009, to appear.

[11] D. Teneketzis and Y. Ho, "The decentralized Wald problem," *Information and Computation (formerly Information and Control)*, vol. 73, no. 1, pp. 23–44, Apr. 1987.

[12] J. C. Walrand and P. Varaiya, "Optimal causal coding—decoding problems," *IEEE Trans. Inf. Theory*, vol. 29, no. 6, pp. 814–820, Nov. 1983.

[13] V. S. Borkar, S. K. Mitter, and S. Tatikonda, "Optimal sequential vector quantization of Markov sources," *SIAM Journal of Optimal Control*, vol. 40, no. 1, pp. 135–148, Jan. 2001.

[14] A. Mahajan and D. Teneketzis, "Optimal design of sequential real-time communication systems," *submitted to IEEE Trans. Inf. Theory*, Jan. 2006.

[15] ——, "On the design of globally optimal communication strategies for real-time communcation systems with noisy feedback," *IEEE J. Sel. Areas Commun.*, May 2008.

[16] A. Nayyar and D. Teneketzis, "On jointly optimal real-time encoding and decoding strategies in multiterminal communication systems," in *proceedings of 47th IEEE Conference of Decision and Ccontrol*, Dec. 2008.

[17] A. Mahajan and D. Teneketzis, "Optimal performance of networked control systems with non-classical infromation structures," *SIAM Journal of Control and Optimization*, vol. 48, no. 3, pp. 1377–1404, May 2009.

[18] A. Anastasopoulos, "Engergy-delay tradeoff in multiple access channels," 2009, submitted.

[19] J. A. Tatman and R. D. Shachter, "Dynamic programming and influence diagrams," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 20, no. 2, pp. 365–379, Mar./Apr. 1990.

[20] H. S. Witsenhausen, "The intrinsic model for discrete stochastic control: Some open problems," in *Control Theory, Numerical Methods and Computer System Modelling*, ser. Lecture Notes in Economics and Mathematical Systems, A. Bensoussan and J. L. Lions, Eds. Springer Verlag, 1975, vol. 107, pp. 322–335.

[21] A. B. Kahn, "Topological sorting of large networks," *Communications of the ACM*, vol. 5, no. 11, pp. 558–562, 1962.

[22] R. D. Shachter, "Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams)," in *proceedings of the fourteenth conference in Uncertainty in Artificial Intelligence*, 1998, pp. 480–487.

[23] D. Geiger, T. Verma, and J. Pearl, "Identifying independence in Bayesian networks," *Networks*, vol. 20, no. 5, pp. 507–534, 1990.

[24] ——, "d-separation: from theorem to algorithms," in *Proceedings on the fifth Workshop on Uncertainity in Artificial Intelligence*, Windsor, Ontario, 1989, pp. 118–125.

[25] H. S. Witsenhausen, "On the structure of real-time source coders," *Bell System Technical Journal*, vol. 58, no. 6, pp. 1437–1451, July-August 1979.

[26] ——, "On policy indepdence of condtional expectation," *Information and Control*, vol. 28, pp. 65–75, 1975.

[27] D. Teneketzis, "On the structure of optimal real-time encoders and decoders in noisy communication," *IEEE Trans. Inf. Theory*, pp. 4017–4035, Sep. 2006.