On the synthesis of useful social laws for artificial agent societies
(preliminary report)

Yoav Shoham and Moshe Tennenholtz
Robotics Laboratory
Department of Computer Science
Stanford University
Stanford, CA 94305

Abstract

We present a general model of social law in a computational system, and investigate some of its properties. The contribution of this paper is twofold. First, we argue that the notion of social law is not epiphenomenal, but rather should be built into the action representation; we then offer such a representation. Second, we investigate the complexity of automatically deriving useful social laws in this model, given descriptions of the agents' capabilities, and the goals they might encounter. We show that in general the problem is NP-complete, and identify precise conditions under which it becomes polynomial.

1 Introduction

This paper is concerned with the utility of social laws in a computational environment, laws which guarantee successful coexistence of multiple programs and programmers.We imagine an environment in which multiple planners/actors are active; we will call these 'agents' here, without attaching precise technical meaning to the term (but see the formal development in the next section).[1]

To illustrate the issues that come up when designing a society, consider the domain of mobile robots. Although still relatively simple, state-of-the-art mobile robots are able to perform several sorts of tasks. They can move from place to place, identify and grasp simple objects, follow moving objects, and so on. Each of these tasks involves sophisticated techniques, but is, broadly speaking, achievable with existing planning and control technology. However, when we consider gathering several robots in a shared environment, a host of new problems arises. The activities of the robots might interfere with one another: The planned spacetime paths of robots might intersect, an object needed by one robot might be removed by another, bottlenecks might occur, and so on. Similarly, robots may require the assistance of other robots to carry out their task.

How is one to deal with these phenomena? There are two extreme answers, neither of which is in general acceptable. One is to assume a single programmer, whose job it is to program all the robots. As such, he will need to worry about all possible interactions among them, in exactly the same way as he worries about the interactions among the different actions of a single robot (see [2] or [5]). This answer is unsatisfactory for several reasons. First, it is unreasonable to expect that a single individual will control all agents. For example, in the Gofer project [3] 100 or so mobile robots are to be programmed individually in much the same way as workstations are used. Second, the set of agents can be expected to change over time, and one would hardly want to have to reprogram all agents upon each addition or deletion of an agent. Finally, even if a single programmer were given the task of programming all the agents, we have given him no guidelines as to how to achieve the task. To the extent that he will proceed by programming each agent individually, he will be forced to deal with the interactions among the different programs.

An alternative extreme answer is to admit that agents will be programmed individually in an unconstrained fashion, to acknowledge that as a result interactions will occur, and to equip the agents with the means for handling these interactions during execution. These means may take various forms. For example, one approach is to merely detect the interactions as they occur, and appeal to a central supervisor for resolution. An alternative method for handling interactions is to equip the agents with communication capabilities, and program them to engage in series of communications to resolve interactions. Again, using

[1] As a special case, we are interested in extending the framework of Agent Oriented Programming (AOP) [10].

the domain of mobile robots for illustration, when two robots note that they are on a collision course with one another, they may either appeal to some central traffic controller for coordination advice, or alternatively they might engage in a negotiation resulting (say) in each robot moving slightly to its right. The use of negotiation to resolve conflicts is common in the distributed AI literature (see [1] or [6]). Nonetheless, there are limitations to this second approach as well. By placing no constraints in advance, the number of interactions may be prohibitive; either the central arbiter may be deluged by pleas, or else agents will have to enter into negotiations at every step. While we certainly do not argue against the utility of either a central coordinator or a negotiation mechanism, we do argue that it is essential to add a mechanism that will minimize the need for either. Again, we draw on the domain of mobile robots for intuition. Suppose robots navigate along marked paths, much like cars do along streets. Why not adopt a convention, or, as we'd like to think of it, a social law, according to which each robot keeps to the right of the path? If each robot obeys the convention, we will have avoided all head-on collisions without any need for either a central arbiter or negotiation.

This then is the image, which is not original to us; it is implicit in many places in AI, and was made explicit already by Moses and Tennenholtz ([7],[8]). The society will adopt a set of laws; each programmer will obey these laws, and will be able to assume that all others will as well. These laws will on the one hand constrain the plans available to the programmer, but on the other hand will guarantee certain behaviors on the part of other agents. The two approaches discussed above simply mark the endpoints of this tradeoff continuum. The first, "single programmer" approach stretches the notion of a social law so as to completely dictate the behavior of each agent, leaving no freedom to the individual programmer. The second approach adopts an opposite, degenerate form of social law, the vacuous law. The goal of the intended theory of social laws will be to relax the restriction to these extreme solutions, and instead to strike a good balance between allowing freedom to the individual programmers on the one hand and ensuring the cooperative behavior among them on the other.

How is one to decide on appropriate social laws? One approach is to hand-craft laws for each domain of application. This is the approach we take in a companion paper [9], where we present a number of traffic laws for a restricted domain of mobile robots. Several systems built by DAI researchers have used organization structures (see [4]) for enabling agents to achieve a cooperative goal. These structures can be considered as a type of social laws. In this paper we tackle a different problem; we are interested in a general model of social law in a computational system, and in general properties that can be proved in that model. The contribution of this paper is twofold. First, we argue that the notion of social law, or constraints, is not epiphenomenal, but rather should be built into the action representation; we then offer such a representation. Second, we investigate the complexity of *automatically* deriving useful social laws in this model, given descriptions of the agents' capabilities, and the goals they might encounter over time. We show that in general the problem is NP-complete, and identify precise conditions under which it becomes polynomial.

The remainder of this paper is structured as follows. In the next section we set up the formal model of multi-agent actions. In section 3 we state the computational problem precisely, and show that in its full generality the problem is NP-complete. We also show a similar result for one natural restriction on the general model. In section 4 we formulate additional natural restrictions on the general model; we show that these conditions are both necessary and sufficient for the problem to become polynomial. Finally, we look at a special case in which the states of agents happen to be encodable concisely, and show that there too the problem is polynomial. We conclude with a summary, discussion of related work, and planned future work.

2 The general framework

The term 'agent' is common in AI, and used in diverse ways. Here we will take an agent to have state, and to engage in actions which move it among states. Although this basic definition is consistent with most uses of the term in AI, it is perhaps somewhat too impoverished to be worthy of the lofty name. However, our goal here is not to contribute to the theory of individual agents but to the theory of social law, and so it will be illuminating to initially adopt a model in which all the complexity arises from the existence of multiple agents, and not from the behavior of individual agents.

We adopt a synchronous model: Agents repeatedly and simultaneously take action, which leads them from their previous state to a new one. The actions of an agent is taken from a given repertoire. The problem in defining the transition functions of agents is due to the fact that the state in which the agent

ends up after taking a particular action at a particular state, depends also on actions and states of other agents. Thus, in principle, we could think of the transition function of the entire system, a mapping from the states and actions of all agents to the new states of all agents. In this view, for example, in order to determine the effect of one car turning left at an intersection, we would have to specify the states and actions of all other cars. An alternative view would be to define the transition function of each agent independently, and account for the effects of other agents by making the function nondeterministic. In this view, for example, the effect of turning left at an intersection would be either a new position and direction of the car, or a collision.

These are two extreme approaches to modelling concurrent actions. In the first approach, *all* information about other actions must be supplied, and the transition function produces the most specific prediction. In the second approach, *no* information about other agents is supplied, and the transition function produces the most general prediction. Instead of adopting either extreme view, we propose an intermediate approach. We propose adding the concept of *social law*, or constraints on the behavior of agents. The constraints specify which of the actions that are in general available are in fact allowed in a given state; they do so by a predicate over that state. The transition function now takes as an argument not only the initial state and the action taken by the agent, but also the constraints in force; it produces a prediction about the set of possible next states of the agent. For example, it might predict that as a result of turning left at the intersection, *given certain traffic rules*, the car will successfully complete the turn.

We claim that this is a natural representation of actions, and an advantageous one. Generally speaking, the prediction based on constraints will be more general than the prediction that is based on the precise states and actions of all other agents, and more specific than the prediction based on *no* information on the other agents.

A final comment, before we present the formal model. We make an assumption of *homogeneity*; specifically, we assume that the sets of states and the available actions are common to all agents. We do not assume that the agents will necessarily be in the same state at any time, nor that they will take the same action when in the same state; only that they "have the same hardware." Similarly, we assume an egalitarian society, in which the same constraints apply to all agents. None of these assumptions are crucial to our approach, but they simplify the discussion.

The formal model

Definition 2.1: Given a set of states S, a first order language $\mathcal{L}$ (with an entailment relation $\models$), and a set of actions A, a *constraint* is a pair (a, φ) where $a \in A$ and $\varphi \in \mathcal{L}$ is a sentence. A *social law* is a set of constraints (a_i, φ_i), at most one for each $a_i \in A$.

The language $\mathcal{L}$ will be used to describe what is true and false in different states. Given a state $s \in S$ and a sentence $\varphi \in \mathcal{L}$, s might satisfy or not satisfy φ. We denote the fact that s satisfies φ by $s \models \varphi$. The intuitive meaning of (a_i, φ_i) will be that φ_i is the most general condition about states which *prohibits* taking action a_i.

In the sequel, we will use the following notation. Given a pair of social laws, sl_1 and sl_2, we denote by $sl_2 < sl_1$ the fact that for every $(a_i, \varphi_i) \in sl_2$ there exists $(a_i, \varphi_j) \in sl_1$ such that $\varphi_j \models \varphi_i$. Intuitively, it will mean that sl_1 is more restrictive than sl_2.

Definition 2.2 : A *social agent* is a tuple $(S, \mathcal{L}, A, SL, T)$ where S, $\mathcal{L}$, A are as above, SL a set of social laws, and T is a total transition function $T : S \times A \times SL \rightarrow 2^S$ such that:

- For every $s \in S, a \in A, sl \in SL$, if $s \models \varphi$ holds and $(a, \varphi) \in sl$ then $T(s, a, sl) = \Phi$, the empty set.
- For every $s \in S, a \in A, sl1 \in SL, sl2 \in SL$, if $sl2 < sl1$ then $T(s, a, sl1) \subseteq T(s, a, sl2)$.

In practice, the transition function T will be only partially specified. If $T(s, a, sl)$ is not explicitly defined for a particular sl, then $T(s, a, sl)$ is assumed to be the conjunction of all explicitly defined $T(s, a, sl_i)$ satisfying that $sl_i < sl$. If this conjunction is over an empty set then $T(s, a, sl) = \Phi$, the empty set.

Definition 2.3: A *social multi-agent system* is a collection of social agents which share the set of states, the language for describing states, the set of potential actions, the set of potential social laws, and the transition function.[2]

[2]Recall again that this only means that the agents "have the same hardware".

3 The Computational Problem

The multi-agent system defined in the previous section provides one degree of freedom in addition to those present in standard multi-agent models: The social law in effect. Once we fix the social law, the social multi-agent system reduces to a standard one, since all transitions which are incompatible with this law are now ignored. (Note, however, that the remaining transitions may still be nondeterministic.) Thus, a social multi-agent system and a particular social law together induce a standard multi-agent system, which makes no reference to social laws. Loosely speaking, the computational problem will be to select from among the candidate social laws one that, given the social multi-agent system, will induce a 'good' standard system. But what makes for a 'good' system?

For this purpose we identify a subset of the set of states, which we call *focal states*. We will be interested in a social law which will ensure that each agent, given two focal states, is able to construct a plan guaranteed to move him from one state to the other – *no matter what actions are taken by other agents*. Note the existence of a certain tradeoff: The more restrictive the law, the more can the planner rely on the effects of his actions, but, on the other hand, the more restrictive the law, the fewer actions are available to the planner in the first place. In selecting a social law we wish to strike a good balance, allowing each agent enough freedom to achieve its goals but not enough freedom to foil those of others.

We now turn to the formal development. Intuitively, an agent's *legal plan* is a decision on how to act in each state, in a way which is consistent with the law in force:

Definition 3.1 : Given a social agent $(S, \mathcal{L}, A, SL, T)$ and a social law $sl \in SL$, a *legal plan* is a total function $DO : S \rightarrow A$ such that if $(a, \varphi) \in sl$ and $s \models \varphi$ holds, then $DO(s) \neq a$. An *execution* of the plan from a state s_0 is a sequence of states $s_0, s_1, s_2, \ldots$ such that $s_{i+1} \in T(s_i, DO(s_i), sl)$.

Note that a plan forces the agent to take action at every step, but we certainly allow the user to include the null action, which might leave states unchanged. Also note that even if the null action is included, some social laws may prohibit it in certain states!

Definition 3.2: Given a social multi-agent system and a subset F of the set of states (the focal states), a *useful law* is a law for which, given any $s_1, s_2 \in F$, there exists a legal plan such that *every* execution of that plan in s_1 includes s_2.

Note that, strictly speaking, we cannot speak of a plan 'leading' to the goal, since the plan is by definition infinite. Also note that this definition does *not* mean that there necessarily exists one *fixed* sequence of actions that will connect two focal states; an action of the agent may have nondeterministic effects, and the following action in the execution of the plan may depend on the state in which the agent landed.

We are now in a position to phrase a precise computational problem:

Definition 3.3: [The Useful Social Law Problem (USLP)] Given a social multi-agent system and a set of focal states, find a useful law if one exists, or, if no such law exists, announce that this is the case.

The technical results in this paper concern the computational complexity of the USLP. In order to present quantitative results we have to be more precise about some of the details of our model. In the following we will assume that the number of states in the representation of an agent is finite and is denoted by n, and we will measure the computational complexity as a function of n. We assume that the total size of an agent's representation is polynomial in n. We also assume that each property of the form $s \models \varphi$, and of the form $sl_1 < sl_2$ can be efficiently verified.

The following theorem shows that the general USLP is intractable, although its complexity is lower than the complexity of many other problems discussed in multi-agent activity (see [11]):

Theorem 3.1: *The USLP is NP-complete.*

We point again that, since the USLP is computed off-line, this result is not entirely negative. Still, it would be satisfying to be able to achieve lower complexity by imposing various restrictions on the structure of agents. This is what we do in the next two sections.

4 Several Restrictions on the General Model

We start by imposing the following restriction: For each state s, the number of transitions which might

change s is bounded by $O(log(n))$.[3] This is a straightforward generalization of the natural "bounded fan-out" restriction which is common for classical automata. Intuitively, this restriction says that the number of actions an agent might perform at a given state is small relative to the total number of states, while the quality of the information about constraints which might be relevant to the effects of a particular action in a particular state is relatively high. Our logarithmic bound enables us to treat the case in which the number of transitions which are applicable in any given state is small relative to the total number of states, while it is still a function of that number. Notice that the total number of actions and social laws appearing in the representation might still be much more than logarithmic.

The computational problem related to the above restriction is defined as follows:

Definition 4.1: [The Bounded Useful Social Law Problem (BUSLP)] Given a social multi-agent system where the number of transitions which might change a particular state is bounded by $O(log(n))$, and a set of focal states, find a useful law if one exists, or, if no such law exists, announce that this is the case.

On it own, this natural restriction does not buy us a whole lot as far as the computational complexity goes:

Theorem 4.1: *The BUSLP is NP-complete.*

However, we have not presented this restriction as a mere curiosity; we now present precise conditions under which the BUSLP become tractable. Consider the following three restrictions:

1. The number of focal states is bounded by a constant c.
2. For any pair of focal states $s_1, s_2 \in F$, there exists a legal plan all of whose executions reach s_2 starting at s_1 *while visiting the same sequence of states.* Intuitively, this requirement states that it is not enough that there be a plan, as required in the definition of a useful law; the plan must be deterministic.
3. For any pair of focal states $s_1, s_2 \in F$, there exists a legal plan all of whose executions reach s_2 starting at s_1 *in no more than* $O\left(\frac{log(n)}{log(log(n))}\right)$ *steps.* Intuitively, this requirement states that it is not enough that there be a plan; the plan must be short.

[3] If $T(s,a,sl_1) = T(s,a,sl_2)$ for $sl_2 < sl_1$ then we do not count $T(s,a,sl_1)$ as one of the transitions.

These three restrictions may or may not be acceptable; although we expect that they are reasonable in some interesting applications, we take no stance on this here. The significance of these restrictions is that they allow us to state precise conditions under which the BUSLP becomes polynomial:

Theorem 4.2: *The BUSLP is polynomial if restrictions 1, 2, and 3 hold, and NP-complete otherwise.*

5 Succinct representation of agents

In the previous section we provided necessary and sufficient conditions under which the BUSLP becomes polynomial; note that this result does not provide necessary conditions for the USLP to become polynomial (it provides only sufficient conditions for that). Indeed, in this section we mention a different restriction of the USLP which guarantees polynomial time complexity.

The general framework imposes no structure on the set of states, and places no restrictions on the number of laws which affect the results of any particular action. In practice, there is much more structure in the makeup of agents. For example, while the set of states might be very large, it is usually possible identify components of the agent, such that the set of states of the agent consists of the Cartesian product of the sets of states of the individual components. If we consider for example the representation of a robot, then one component may relate to its location, another to its orientation, and another to its arm position.

This modularity of states is the first restriction we consider here; specifically, we assume that there exist $O(log(n))$ components, each of which can be in one of a constant number of states. Thus the state of an agent consists of the Cartesian product of these states. The total number of an agent's states is still n; note that this is no contradiction.

The modularity of state gives rise to a modularity of action; usually, only a small number of social laws will be relevant to determining the change in the state of a particular component as a result of a particular action. In the robotic setting, for example, a change in the particular location of the robot might depend on particular traffic laws, but not on laws requiring one to shut windows after opening them, on laws prohibiting one from taking the last cookie

without announcing the fact, and perhaps not even on many of the possible traffic laws. We capture this intuition by requiring that the change of a particular state of a particular component can depend on only a constant number of social laws.

We will use the term *modular social agent* to denote an social agent whose structure has these two properties of modularity, and *modular social multi-agent system* to denote a collection of such agents.[4] Notice that these restrictions are separate from those discussed in the previous section. Modular systems have the following property:

Theorem 5.1: *Given a modular social multi-agent system, the USLP is polynomial.*

6 Conclusions

A basic approach to coordinating multiple agents is to restrict their activities in a way which enables them to achieve their dynamically-acquired goals while not interfering with other agents. Although this is, we believe, a commonsensical approach, it has not received much attention so far within AI; exceptions include [7] and [8], where the authors investigate several basic principles and formal aspects of this approach. We have presented model of multi-agent action whose novel feature is the explicit mention of social laws in the definition of agents. We then investigated the fundamental computational problem involved with finding useful social laws: We showed the general problem intractable, and then identified restrictions which achieve tractability. We believe that the formal model and complexity results both constitute an essential contribution to our understanding of the role of social law in artificial intelligence.

Much remains to be done. For example, as was mentioned at the beginning, we have assumed a homogeneous society, both in term of the makeup of agents and in terms of the applicability of social laws. In future work we will relax this assumption, made here only for convenience. Harder assumptions to relax include the assumption of law-abiding citizenship; here we have assumed that all agents obey the social laws, but what happens if some don't? How vulnerable is the society to rogue agents? In fact, arguments had been made about that social laws will sometimes be break, and that the penalty for this should be considered in the stage of design. Similarly, we have assumed that the useful social law is computed off-line. Sometimes this is infeasible, however, either because the makeup of the agents is not known in advance, or because it changes over time. In such cases, can we devise methods by which, through trial error, the agents will over time converge on a social law? These are some of the questions in which we are currently interested.

[4] A full formal definition will appear in the full paper.

References

[1] A. H. Bond and L. Gasser. *Readings in Distributed Artificial Intelligence.* Ablex Publishing Corporation, 1988.

[2] S.J. Buckley. Fast motion planning for multiple moving robots. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, pages 322–326, 1989.

[3] P. Caloud, W. Choi, J.-C Latombe, C. Le Pape, and M. Yim. Indoor automation with many mobile robots. In *Proceedings IEEE International Workshop on Intelligent Robots and Systems, Tsuchiura, Japan*, 1990.

[4] Edmund H. Durfee, Vicror R. Lesser, and Daniel D. Corkill. Coherent Cooperation Among Communicating Problem Solvers. *IEEE Transactions on Computers*, 36:1275–1291, 1987.

[5] M. Erdmann and T. Lozano-Perez. On multiple moving robots. *Algorithmica*, 2(4):477–521, 1987.

[6] S. Kraus and J. Wilkenfeld. The Function of Time in Cooperative Negotiations. In *Proc. of AAAI-91*, pages 179–184, 1991.

[7] Yoram Moses and M. Tennenholtz. Artificial Social Systems Part I: Basic Principles. Technical Report CS90-12, Weizmann Institute, 1990.

[8] Yoram Moses and M. Tennenholtz. On Formal Aspects of Artificial Social Systems. Technical Report CS91–01, Weizmann Institute, 1991.

[9] Y. Shoham and M. Tennenholtz. On Traffic Laws for Mobile Robots. Submitted to AIPS-92.

[10] Yoav Shoham. Agent Oriented Programming. Technical Report STAN-CS-1335-90, Dept. of Computer Science, Stanford University, 1990.

[11] M. Tennenholtz and Yoram Moses. On Cooperation in a Multi-Entity Model. In *Proc. 11th International Joint Conference on Artificial Intelligence*, 1989.