ELSEVIER

Artificial Intelligence 119 (2000) 61–101

Artificial Intelligence

www.elsevier.com/locate/artint

Choosing social laws for multi-agent systems: Minimality and simplicity ☆

David Fitoussi [1], Moshe Tennenholtz *

Faculty of Industrial Engineering and Management, Technion—Israel Institute of Technology, Haifa 32000, Israel

Received 15 February 1999

Abstract

The design of social laws for artificial agent societies is a basic approach to coordinating multi-agent systems. It exposes the spectrum between fully-centralized and fully-decentralized coordination mechanisms. Useful social laws set constraints on the agents' activities which allow them to work individually in a mutually compatible manner. The design of useful social laws is a problem of considerable importance. In many cases, several useful social laws might be considered, and we might wish to have some criteria in order to choose among them. In this paper, we present the notions of minimal and simple social laws, which capture two basic criteria for selecting among alternative (useful) social laws, and study these criteria in the framework of basic settings, namely Automated Guided Vehicles and Distributed Computing. We also present results with regard to computational issues related to minimal and simple social laws, and to the relationship between these two concepts. Together, the new insights provided here can be used as a basic framework for the analysis of "good" social laws, and initiate research on the selection among alternative social laws.

Keywords: Artificial social systems; Minimal social laws; Simple social laws

1. Introduction

The design of an agent which is to operate in a multi-agent environment [5,9,12,13,16] is quite a different task from the design of an agent which performs its activities in isolation

☆ Part of the results on minimal social laws appears in the paper "Minimal social laws", in: Proceedings AAAI-98, Madison, WI, 1998.

* Corresponding author. Email: moshe@robotics.stanford.edu; moshet@ie.technion.ac.il.

[1] Email: quartz@mit.edu.

0004-3702/00/$ – see front matter
PII: S0004-3702(00)00006-0

from other agents. While in the latter, we are mainly concerned with providing the agent with the capacity and appropriate knowledge, which are required for deciding on a suitable course of action, in multi-agent settings we must deal with the coordination of several agents.

In this research, we consider a particular approach to coordination, referred to as the artificial social system approach [1,6,38,39,42,44,56,61]. An artificial social system institutes a social law which the agents shall obey. Intuitively, a social law restricts, off-line, the actions legally available to the agents, and thus minimizes the chances of an on-line conflict, and the need to negotiate. Similarly to a code of laws in a human society [50], an artificial social law regulates the individual behavior of the agents and benefits the community as a whole. Yet, the agents should still be able to achieve their goals, or any other specification of the system, and restricting their legal actions to a too wide extent might leave them with no possible way to do so.

Meeting the system specification is thus a *sine qua non* condition for accepting a candidate social law, but is it the ultimate criterion? Should we be satisfied once the law we designed guarantees this specification (provided every agent conforms to the law)? Clearly, there might still be quite a diversity of laws with such a characteristic. In order to compare useful social laws, we will need to define ways to rank these laws, and that is the purpose of this study. Our aim is to present guidelines for selecting suitable social laws from among a set of possible (useful) social laws. In order to do so, we define and investigate criteria that provide a way for ranking alternative social laws.

The first criterion that we consider is called minimality. When considering a potential social law, we ask ourselves whether this is the *minimal* set of constraints needed to be obeyed by the agents in order to guarantee the goal specification. In other words, minimal social laws attempt to minimize the amount of constraints set on the agents and as such, captures the notion of maximal individual flexibility.

This criterion is quite general and can be applied to a variety of settings as we demonstrate by studying the characteristics of different social laws in two domains of interest. The first domain relates to robotics where we are concerned with many automated guided vehicles (AGVs) [24,33] moving together on a circular track in order to achieve given goals. The AGVs must solve a motion planning problem, i.e., find a path leading them from their current location to another one. Our goal as designers is to guarantee a collision-free environment for all the vehicles whereas, for an AGV taken individually, this system-level goal might conflict with its wish to be granted maximal freedom. Minimality will be shown helpful in guiding the designer in his task of finding an appropriate social law for this system. The second example is taken from the distributed computing literature. It is a classical problem that addresses agreement (consensus) among distributed processes in a communication network [18,19,43]. As with the robotics environment, minimal social laws are designed and analyzed. After studying minimal social laws in these domains of interest, we turn our attention to computational studies of minimal social laws, and deal with the complexity of the automatic synthesis of minimal social laws. We analyze the problem of relaxing a given social law by removing constraints. We show that this problem is NP-hard; the proof of this result uncovers connections of this problem with the problem of planning with incomplete information. In addition, we show that, in a restricted setting, efficient synthesis of minimal social laws is achievable.

Although minimality appears to be helpful in designing flexible social laws, it is not to be seen as an ultimate criterion. Therefore, we consider an alternative criterion, termed simplicity. Simple laws directly relate to the capabilities the agents need to be equipped with in order to abide by the law. The simpler the social law, the easier it is for simple agents to follow it. Simplicity brings forth new computational issues. We formally define this concept and discuss some of its properties. In addition, we investigate the computational complexity of designing some specific sub-classes of simple social laws. As with minimality, this problem is, in general, computationally hard, but can be efficiently solved in more restricted settings.

Finally, we consider connections between minimality and simplicity. These connections are useful as they allow us to grasp the intricacies of each concept. In general, these criteria are loosely related, though again, in some cases, strong connections can be shown to exist. In particular, we show that a nontrivial efficient law introduced in previous work on artificial social systems is both simple and minimal.

Looking at criteria for choosing social laws is an important research issue. Such criteria are likely to provide designers of a system with useful guidance as to which law to impose on the system. Minimality captures the intuition that while the imposition of social restrictions is useful, it should be done only to the extent it is necessary. The important point is not that all systems will necessarily benefit from minimal social laws. In some systems in which flexibility is of little concern, one might prefer to let the agents with minimal choice. However, minimality makes us aware of dimensions that should be explored when attempting to tailor a social law to the needs of a system. In some cases, other dimensions, like simplicity may have to be explored as well, and sometimes might be preferred. Thus, the main contribution of this paper is to provide a framework for engineering social laws in systems (taking into account various design principles), where minimality and simplicity play a central role.

This paper is structured as follows. In Section 2 we present the notion of *minimal* social laws, as a basic criterion for choosing among useful social laws. In Section 3 we apply this concept in the context of two basic domains: automated guided vehicles moving in a ring environment, and consensus problems in distributed systems (these two domains demonstrate the generality of the approach across different basic settings). In Section 4 we present a computational study of minimal social laws. We show that a basic problem in the design of minimal social laws is intractable. However, in Section 5 we point to a setting where a polynomial time algorithm for the synthesis of such a law is presented. In Section 6 we present the notion of *simple* social laws, as a second basic criterion for choosing among social laws, and in Section 7 we study some of its computational properties. In Section 8 we present and discuss several results on the integration of minimality and simplicity. Section 9 concludes with an additional discussion of related work and of the contributions of this paper.

2. Minimal social laws

A problem that we believe to be essential for the design of artificial social systems is the search for an optimal law, according to some measure of optimality. With this purpose

in mind, we shall now present and investigate the notion of minimal social laws. This section presents a general model where this concept/criterion can be studied. In Section 3 we illustrate the use of this criterion in two domains of interest. Sections 4 and 5 consider computational and algorithmic issues related to the design of minimal social laws.

2.1. A design criterion for optimality

A central issue in the design of social laws which has not been investigated yet is the design of optimal social laws, according to some measure of optimality. Given a system of agents and a specification, the job of the designer is to find an implementation consistent with this specification. A possible way to come up with a suitable implementation could be to identify a strategy profile such that, given that each agent acts according to the strategy assigned to it in this profile, the specification is satisfied. Then, it would be possible to enforce this behavior by instituting the appropriate behavior as the law of the system [37, 38]. Clearly, this law would be consistent with the system specification (according to which it has been designed), since it would force each agent to use the strategy prescribed to it in the strategy profile, while this strategy profile has been chosen to satisfy the specification of the system. Most often, other useful social laws will exist, and we will be faced with the problem of comparing the different laws. How should we do so? When could we say that we have found an optimal law, and under which criterion? Following the literature on mechanism design in economics [21], a law could be considered optimal if it brings maximal utility to the designer of the system. Although legitimate, this definition does not capture the fact that the purpose of social laws is to provide a flexible framework for a system to evolve in. This will therefore motivate another notion of optimality, that we call minimality, and which we would like to relate to the impact social laws have on the dynamics of the system (and its components). Optimality would then be seen as a maximization of the agents' ability to adapt to unpredictable changes in the environment, and in their goals, subject to the requirement that they be able to obtain their original goals.

It is first necessary to understand what is meant by comparing two laws. Informally, given two different useful laws l_1 and l_2, we say that l_2 is smaller than l_1 if the set of behaviors induced by strategies consistent with l_1 is included in the set of behaviors induced by strategies consistent with l_2. In other words, if we regard a law l as a set of restrictions, l_2 sets *less* restrictions than l_1. Intuitively, a smaller law is a law that rules out less behaviors consistent with the agents' goals. We relate the size of a law to optimality in the following way: A useful social law l^* is minimal (and optimal) for some system specification, if and only if, for any other useful social law l, l is not smaller (as defined above) than l^*.

A minimal law will grant the agents more freedom in the process of choosing their behavior while ensuring that they conform to the system specification. Systems with smaller social laws are therefore likely to be more robust to changes in the environment specification or in the capabilities of the agents. A small social law might also be more easily enforced, as being less restrictive, especially in environments where enforcement of the law need to rely also on rationality constraints. Note also that minimal laws need not

be unique, and selecting one law among several minimal social laws might require some other exogenous criterion (either quantitative or qualitative).

2.2. Formal definitions

The following definitions formulate the notions of social laws and minimality in the framework of a general strategic model. For ease of exposition, we present our definitions for environment consisting of two agents. Extension to the case of n agents ($n \geqslant 2$) follows easily. [2]

Definition 2.1. An environment E is a tuple $\langle N, S_1, S_2 \rangle$, where $N = \{1, 2\}$ is a set of agents and S_i is a set of strategies available to agent i.

Given an environment, the agents are assigned goals that they must obtain by devising appropriate strategies from the set of possible strategies available to them. In addition there are some system-level goals that should always be guaranteed.

Definition 2.2. In an environment $E = \langle N, S_1, S_2 \rangle$ a goal g is a subset of the Cartesian product over the agents' strategy spaces, i.e., $g \subseteq S_1 \times S_2$.

The above definition captures goals in very general terms. Roughly speaking, a goal is associated with the set of joint strategies in which it is indeed obtained. Notice that this broad definition of goals enables us to refer to complex goals, e.g., if the strategies are temporal policies then goals would be temporal as well,

We distinguish between several sets of goals. Let us denote by G_i the set of *liveness goals for agent* i. These are goals that we wish to enable agent i to obtain. Naturally, at a given initial state the agent may wish to obtain a particular goal and another goal may be irrelevant. [3] In addition, there is a set G_{safe} of *safety goals*. These are goals that should always be obtained. The formal definition of goal achievement for liveness and safety goals is given below.

The main purpose of (useful) social laws is to set regulations that ensure safety and enable the agents to obtain their liveness goals. Given an environment and a goal for agent i, it is not certain that this agent has a strategy such that, independently of the strategy profile chosen by his fellow agents, he will achieve his goals. The job of the designer is to devise a social law that restricts the agents' activities. Ensuring that each agent in the resulting system achieves his goals will qualify this social law as useful.

[2] Extension to the case of n agents is straightforward. In fact, the n-agent case can be regarded as a 2-agent problem, where for each agent i, the "other" agent is the group of the $n-1$ remaining agents. The set of strategies of this group is the cartesian product of the strategy space of the individual agents in the group, and the goal set of the group is the union of the individual goal sets. This is similar to the use in game theory of the notation s_{-i} to represent the strategies of all the players except i.

[3] The initial state is implicit in the agent's strategy in our general model. It will be treated more specifically in our more concrete applications and computational study.

Definition 2.3. Given an environment $\langle N, S_1, S_2\rangle$, and given the sets of goals G_1, G_2, and G_{safe}, a social law is a set of restrictions $SL = \langle \overline{S_1}, \overline{S_2}\rangle$ such that $\overline{S_1} \subseteq S_1$ and $\overline{S_2} \subseteq S_2$. SL is *useful* if:

(1) for every goal $g_{1_i} \in G_1$ there exists $s_{1_i} \in S_1 \setminus \overline{S_1}$ such that for all $s_2 \in S_2 \setminus \overline{S_2}$ we have $(s_{1_i}, s_2) \in g_{1_i}$.
(2) for every goal $g_{2_i} \in G_2$ there exists $s_{2_i} \in S_2 \setminus \overline{S_2}$ such that for all $s_1 \in S_1 \setminus \overline{S_1}$ we have $(s_1, s_{2_i}) \in g_{2_i}$.
(3) for every $g_j \in G_{safe}$ and for all $s_1 \in S_1 \setminus \overline{S_1}$, $s_2 \in S_2 \setminus \overline{S_2}$, we have that $(s_1, s_2) \in g_j$.

Following a social law corresponds to choosing strategies within the set of lawful behaviors allowed by the law. It should be pointed out that the mapping between state and action might not always be available, for example due to failures in sensors or lack of access to the state. As a consequence an agent might break the law and the specification cannot be guaranteed. This is an important issue that we intend to study in subsequent work.

As we have discussed before, social laws differ in their properties, and some (useful) social laws are more stringent than others. Our approach to the selection among social laws is to prefer those that satisfy the safety and liveness conditions with minimal constraints. Formally,

Definition 2.4. Consider an environment with a specification of liveness and safety goals. A useful social law $SL = \langle \overline{S_1}, \overline{S_2}\rangle$ is *minimal* if there is no other useful social law $SL' = \langle S'_1, S'_2\rangle$ that satisfies $S'_i \subseteq \overline{S_i}$ for all i.

In the sequel we will refer, unless stated otherwise, to minimal useful social laws as *minimal social laws.*

Minimal social laws give each agent maximal freedom in changing his behavior (this change might be caused by changes in the agent's capabilities, change of goals, etc.). Needless to say, sometimes, changes in the system and new requirements may require a re-design of the social law. The role of minimal social laws is to serve as a basic optimization tool in between these transitions. Given the importance of this type of flexibility, and the need to study the effects of replacing simple protocol design by the design of minimal social laws, we now turn to a study of minimal social laws in two basic domains.

3. Minimal social laws for domains of interest

In the previous section we presented an abstract and general notion of minimal social laws. In the next section we will present a more concrete formal definition in the context of a computational model. In order to make the ideas more concrete, we consider in this section two domains of interest for which we design (minimal) social laws. The first domain is drawn from robotics, and the second from the distributed computing literature. We believe that these domains are quite representative of the use of social laws in the context of physical and software agents, respectively.

3.1. A case study: AGVs in a circular automated assembly line

3.1.1. The domain

In a single-robot automated assembly line, a robot is programmed to perform some activity which will lead it to a goal state, i.e., a situation where its goal has been fulfilled. When several agents are acting together, interactions between the agents' actions may tamper with normal operation. Let us consider for example a domain consisting of a simple automated assembly line, where m robots can move between n stations in a circular fashion ($2 \leqslant m \leqslant n$). This domain is represented by a connected undirected graph $G = (V, E)$ where $|V| = n$, $|E| = n$ and for all $v \in V$, $\deg(v) = 2$ (where $\deg(v)$ is the degree of the vertex v), i.e., the domain is represented by a graph with n vertices and a ring topology. Each node in the ring represents a station. In our simple model, a robot can move at time t (we will assume that time steps are discrete and that time is infinite) from the station it stands at to one of its two neighbors, or stay immobile. All the robots move at the same speed, and a robot who left a station at time t will reach one of the adjacent stations at time $t + 1$. In a case where several robots are in the same station then we have a collision. Finally, we will assume knowledge of the immediate environment, in the sense that each robot can observe the state (occupied or free) of the two stations following it (in clockwise order), and the two stations preceding it (in clockwise order). The system may be initialized in any situation, as long as no pair of robots lie on the same coordinate. In order to simplify the discussion, when referring to the directions of movement we will use the terms 'clockwise' and 'forward' interchangeably. Similarly, we will use the terms 'anti-clockwise' and 'backwards' interchangeably.

3.1.2. The motion planning problem

The specification of the system consists of liveness and safety goals. A liveness goal specifies a particular station to be reached. We will assume that any of the stations can be the target of such a goal. The safety goal requires that collisions be avoided. We will now consider a simple social law for obtaining the above-mentioned specification.

Traffic Law 1. *Each robot is required to move constantly clockwise, from one station to the other along the ring.*

It is easy to show that the following holds:

Proposition 3.1. *Traffic Law* 1 *guarantees that no collision will occur and that each robot will reach any location it might want to get to in* $O(n)$ *steps.*

Traffic Law 1 is attractive as it clearly defines what to do in each situation. However, it might be rather constraining. Traffic Law 1 does not leave any choice to the agent when selecting its actions. His behavior resolves to that of a pure-reflex agent. Notice that from a design perspective it is not enough to require usefulness (that is, meeting the system requirements) from a law, since a useful law might implement the specification in a too much constraining way. There is no need to put a particular restriction on an agent, if this

restriction does not interfere with goal achievement by other agents or might lead to unsafe situations. Hence, we need to go further and examine the notion of a minimal law.

Note that our study fits nicely to the general model presented in the previous section. Strategies are built using the three basic actions a robot can take at each station (move clockwise, anti-clockwise, or rest). The state of an agent consists of its recent location/observation and its history, and a strategy for an agent will be a function from its state to action. Goals involve getting from one station to another station on the ring (a liveness goal) as well as avoiding collisions when moving around (a safety goal).

We will be interested in relating minimal laws to optimal behavior. It is clear that we must trade-off between the need to rule out certain kinds of behaviors in order to avoid collisions, and the need to restrict our agents as little as possible leaving them with enough freedom to achieve their goals. The *Golden Mean Problem* in artificial social systems [42] states just that: find a social law that restricts the possible behaviors of the various agents enough to serve all the agents in a good manner. The minimal social law problem on the other hand focuses on finding a social law driven by the specification of the system, but whose dependency on this specification is as loose as possible. The two problems are clearly related. We emphasize that the less we restrict our agents, the more behaviors we leave available, allowing a less stringent and deterministic framework for them to evolve in. Above all, we achieve a level of flexibility, especially in settings with imperfect information where we can not rely on the (weak) assurance provided by the analysis of a situation whose basic components (e.g., the utility values) are possibly erroneous, or evolve with time.

In our simple automated assembly model, we can improve the useful law by relaxing some of the constraints imposed on the behavior of the robots. Note that this is made possible by using knowledge (in that case, knowledge of the surroundings) available to the robot by the time it makes its decision.

Traffic Law 2.

(1) *Staying immobile is forbidden if the station which can be reached by a single anti-clockwise movement is occupied.*
(2) *Moving anti-clockwise is allowed only if the two stations which can be reached by moving anti-clockwise* twice *are free.*

Proposition 3.2. *Traffic Law* 2 *is a minimal* (*and useful*) *social law.*

Proof. Consider the possible situations that occur in the system and the response of the robots when following the law. A robot x might collide with a robot y which is at a distance of one station in front of it, if and only if x rests or moves forward while y moves backwards, or if x moves forward while y rests or moves backwards. However, since in this case y must move forward, we get that collisions will be prevented. If y is in a distance of two stations in front of x then no collision will occur since in this case y can not move backwards. This will guarantee safe behaviors. Since the law allows our robot to always move forward, it can always achieve its liveness goals. Suppose that Traffic Law 2 is not minimal. Therefore, there exists a smaller law, Traffic Law 3, which allows additional behaviors and still meets the system specification. For Traffic Law 3 to

be smaller than Traffic Law 2, it should allow moving backwards when a robot stands two stations behind (in the anti-clockwise direction), or to rest/move backwards when a robot stands one station behind. This implies that in order to be useful Traffic Law 3 should restrict the robots' behavior when moving clockwise, otherwise collisions could occur. But then, it will restrict the whole system at least as much as Traffic Law 2, thus Traffic Law 3 is not smaller than Traffic Law 2. □

3.2. Consensus in a synchronous distributed environment

We now take a look at another basic domain where minimal social laws may play a significant role. We consider a variant of a classical distributed systems problem, the consensus problem. Other variants of it can be treated similarly. This case study holds an additional interest as it illustrates how roles (in this case, the roles of general and lieutenant) can be assigned to agents and be made an integral part of the social law.

3.2.1. The domain

We assume that the distributed system consists of n processes, interconnected by communication hardware, which exchange information by broadcasting one-bit messages. Naturally, an agent may also refrain from broadcasting a message at a particular point. Message passing is assumed to be synchronous (see [59]). We use a pulse model to simulate the synchrony of the system (see [54] for the description of this model in the context of distributed election algorithms). In a pulse model, time is partitioned into intervals of fixed length (termed pulses), such that at each pulse a process receives the messages sent in the previous pulse, performs internal computations, and sends new messages. Besides sending and receiving messages, a process can trigger internal events. The set of internal events of a process always includes a decide event, defined as follows. Each process has an internal *decision* variable, and we say that a process decides on a value a, $a \in \{0, 1\}$, when it sets its decision variable to a. The system is initially in an initial configuration, which is completely determined by the values of the decision variables of the processes. We associate the initial state of each process p with the initial value of its decision variable.

In the sequel, we will assume that the processes communicate by broadcasting messages, and we will use the term 'broadcast' to represent a multiple-send event consisting of n individual and simultaneous send events with the same source s, the same message m, and different destinations (note that if i broadcasts a message then it will receive it as well). The behaviors of the system are further restricted to those executions where no process ever fails.

3.2.2. The consensus problem

Given a time (i.e., pulse number) $T \geqslant 1$, the consensus problem requires that the decision variables of all agents at time T will be identical. This property is the classical *agreement* requirement [18]. This requirement is usually augmented with a *nontriviality* requirement that rules out trivial decision rules, such as "always decide 0". We will use the following nontriviality requirements:

(1) At least one message should be sent until time T.
(2) Consider the first round where a message has been sent. Then, if the initial state of each of the processes which sent a message on that round is v, then at time T the value of the decision variables of all the processes should be v.

This requirement somewhat resembles the one used in Byzantine agreements [18,43]. The intuition behind it is that the values of the processes which have sent their messages first should be weighted more heavily (and in our case should be given full and exclusive priority), since these processes have expressed an acute interest in the decision process by sending their values first whereas the other processes have not. In any case, other nontriviality conditions can be treated similarly.

Consensus problems arise naturally in situations where distributed entities must take a decision whose impact depends on the decision taken by the other "participants". For example, firms in an oligopoly (organized in a cartel) must coordinate their production level in order to keep profits high [30]. Failure to achieve consensus might have drastic consequences (price war). Likewise, in the context of distributed databases, committing modifications made to local replica and updating the main database rely on a consensus phase (Two-Phase Commit protocol [23,31]) between the distributed components of the database.

In terms of the model described in Section 2, we have that a strategy in our setting is a function from a process history (including its initial state, and its history of send/receive events) to actions (messages to be sent and decision to be made). It is straightforward to define agreement and nontriviality as safety goals. Notice that there is no need to require the obvious liveness goal that tells each agent to decide at the end, since we assumed a decision is made by assigning a value to the decision variable.

Given the above setting we are interested in finding social laws that will satisfy the system specification, will be flexible, and will allow the agents maximal freedom in choosing their actions.

Consensus Law 1. *Each process is assigned a label "general" or "lieutenant", such that exactly one process is assigned the label "general". The general is required to decide on a value* $\boldsymbol{a}$ *and to broadcast this value in the first round. The other agents should refrain from sending a message in the first round, and upon receiving the value from the general they should decide on this value. No changes of decision are allowed in the system from this point on.*

It is easy to show that if all processes abide by the above law, then agreement and nontriviality are satisfied. Thus, Consensus Law 1 is useful. This law however is in fact a very restricting protocol. In many cases agents may wish to take various decisions based on various messages received from other agents. In particular, assigning a value to the decision variable may reflect the performance of a particular type of action. The consensus problem has required that at time T the actions to be performed will be identical. However if an agent (or a subset of the agents) may wish to take various decisions (based perhaps on messages sent by other agents) in the meantime, there is no need to constrain it in this regard. Moreover, it seems that Consensus Law 1 does not capture the "essence of the specification". The agreement and nontriviality requirements suggest that we are not

only seeking agreement, but also that the value agreed upon must be mostly influenced by the agents "who care the most about the decision" (i.e., submit messages first), and our protocol does not capture this situation. Notice that these observations are *not* part of the specification; if they were, Consensus Law 1 would become non-useful. This leads yet again to the problem of designing a minimal social law for the related setting, which would enforce the specification with minimal restrictions and capture (at least in a sense) its "essence" in an accurate way.

Consensus Law 2. *Each process is assigned a label "general" or "lieutenant", such that exactly one process is assigned the label "general". If a process sends a message before it has received any message, then it must send the value of its initial state as the content of its message. If at time* $T-1$ *the general has not received any message then it should broadcast the initial value of its decision variable. By time* T *all agents set their decision variable to the value of the majority of the values received in the first pulse a receive event occurred* (*where* 0 *is the default value for the case of a tie*). *In case their decision variable already holds this value, they can refrain from setting it.*

Proposition 3.3. *Consensus Law* 2 *is a minimal* (*useful*) *social law for the consensus problem.*

Proof. In the last round the decision variable of each process is set to the majority of the values received in the first pulse (say, pulse i) a receive event occurred in this process. Given the failure-free communication media all the processes will receive the corresponding message simultaneously. These observations imply that agreement is satisfied. The fact that the processes are required at first to send their initial values, the fact that the majority of the first submitted values is taken at the end, and the fact that if no message has been received until time $T-1$ then the general must send a message, guarantee that nontriviality is obtained. In order to prove that the law is minimal, consider another law, Consensus Law 3, which is still useful but allow additional strategies. This law cannot allow the general to refrain from sending a message at round $T-1$ (if no message has been received by that time), since this may lead (if no additional constraints are put) to a situation where no messages are sent. Similarly, this law cannot allow a process to send a message that is different from its initial value before receiving any message, since this may prevent nontriviality (if no messages are sent in the corresponding round, and no additional constraints are put). Note that the fact we used a majority decision criterion is in no way restricting. Other criteria might do just as well, provided they generate a common decision value for the processes in the system, but not better. Indeed, a decision criterion is a function that maps inputs (messages received in the pulses up to T) to a single output 0 or 1. For a decision rule to be different from the majority principle, it will necessarily have to rule out, for at least one input, a behavior which matches the majority rule. Thus, a social law based on it will not be smaller that Consensus Law 2. Altogether, we get that Consensus Law 3 cannot allow more strategies and Consensus Law 2 is minimal. □

Looking for minimal social laws is therefore helpful in evaluating alternative social laws and choosing among them. It is by no means the only way to do so. We now examine a

more restricted definition of minimality, minimally preserving social laws, and study the relationship between minimality as presented before and minimally preserving social laws. In Sections 6 and 7 we will consider another approach, based on a notion of simplicity, which is not related to minimal constraints.

3.3. Minimally preserving social laws

We now consider a refinement of the definition of minimal social laws given previously. By doing so, we wish to gain a deeper insight into the concept, as well as encourage discussion of the definition itself.

Let us consider an environment where a social law has been set. We assume that the social law is useful, i.e., there exists a legal strategy (that will be called a useful strategy), for each agent, which guarantees the realization of each agent's goal for any strategy chosen by the other agent (provided it is allowed by the law). Given a social law, we denote the set of useful strategies for agent i, induced by the law, by S_i^u. Note that in general there could be more than one strategy in S_i^u and the cardinality of this set for different i need not be the same (albeit always at least one). We incorporate a condition over S_i^u into the definition of minimal social laws in the following way: given two different social laws σ and σ', $\sigma \prec \sigma'$ if $S_i'(\sigma') \subseteq S_i(\sigma)$, where $S_i(\sigma)$ (respectively $S_i'(\sigma')$) is the set of possible strategies induced for agent i by σ (respectively σ'), for all i, and $S_i'^u$ is preserved for all i (i.e., any useful strategy allowed by σ' remains useful when σ is applied to the system). Formally:

Definition 3.4. Given an environment $\langle N, S_1, S_2\rangle$, and given the sets of goals G_1, G_2, and G_{safe} a social law $\Sigma = (\Sigma_1, \Sigma_2)$ with sets $\{S_i^u(\Sigma)\}_{i\in N}$ of useful strategies is minimally preserving if there is no other useful social law $\sigma = (\sigma_1, \sigma_2)$ with sets of useful strategies $\{S_i^u(\sigma)\}_i$ such that $\sigma_i \subseteq \Sigma_i$ and $S_i^u(\Sigma) \subseteq S_i^u(\sigma_j)$ for all i.

The same advantages that applied to minimality motivate the study of minimally preserving social laws. However, instead of expanding the set of legal strategies without other discrimination than keeping the social law useful, minimally preserving social laws emphasize the strategies we are most interested in, the useful ones. Standard minimality makes each agent immune to changes in the goal structure or capacities of other agents, and provides each agent with a larger choice of strategies from which to derive an appropriate strategy for the new configuration. Minimally preserving laws, on the other hand, strives to provide the agents with several alternatives with which each agent can achieve its original goal and also new ones. Minimally preserving social laws are conservative: a particular subset of the legal strategies (namely the useful strategies) is given special status. Very often, this set corresponds to some basic behaviors that we want our system to display. For example, in the AGVs case study above, one such basic behavior is given by Traffic Law 1 and corresponds to the robots moving constantly clockwise. We wish to keep this behavior even when we eventually get to allow many more strategies. Another observation which justifies the study of minimally preserving social laws relates to the issue of designing efficient plans. Expanding the set of strategies is no panacea, and should be done carefully, as we usually introduce more complex (and less efficient) behaviors. By considering

minimally preserving laws, the likely-to-be-simpler useful strategies in a stringent law will remain useful in any subsequent extension of the law. For instance, in the context of the AGVs referred to above, the behavior of moving constantly clockwise is one of the simplest available, and should be preserved. We will return to the issue of simplicity in the following sections, when dealing with simple social laws.

The following observation makes the relationship between the two definitions of minimality given above more precise. In particular, we observe that minimally preserving social laws are not necessarily minimal.

Proposition 3.5. *Minimally preserving social law and minimal social laws are not equivalent.*

(In the proof, we refer to a minimal social law as an MSL and to a minimally preserving social law as an MPSL.)

Proof. It is easy to see that MSLs are necessarily MPSLs: if σ is an MSL, then no extension of σ (in the sense defined above), is an MSL, and therefore no extension of it is an MPSL. Therefore, σ is an MPSL. The converse is not true: consider a two-agent encounter where each agent has two strategies a and b, and the joint strategies $(a, a), (a, b), (b, b)$ all lead to a state where each agent's goal is achieved whereas the joint strategy (b, a) leads each of them to a forbidden state. A possible useful social law might authorize strategy a to agent 1 and both a and b to agent 2. Notice that in this case, strategy a for agent 1 and strategy a for agent 2 are useful strategies (as is strategy b for agent 2). This law is useful, minimally preserving (as adding b to agent 1 would not preserve the useful strategy a of agent 2), but is not minimal (adding b to agent 1 still leaves each agent with one useful strategy, namely (a, b)). □

Having defined the concept of minimality and illustrated its applicability in domains of interest, we now explore computational issues related to the design of minimal social laws.

4. A computational study of minimality

This section, as well as the following one, departs from the very general model presented above and presents a study of minimal social laws in a computational model. This model is useful to address computational issues related to the automatic design of minimal social laws. The process ultimately leads to a characterization of minimality in computational systems and exposes the more abstract description of the previous sections in computational terms, bridging the gap between the theory of minimality and its use in system design. We wish to emphasize that we see the design of social laws, and minimal social laws in particular, as an off-line activity. This implies that the automatic synthesis of such social laws, its considerable importance notwithstanding, is not the ultimate way to design them. The concept of minimality is already interesting when designing specific systems such as those discussed in the previous sections. Nevertheless, a computational study of minimal social laws can still shed light on their design and on their connections with related concepts and issues.

Definition 4.1. A (two-agent) *system* is a tuple $S = \langle L_1, L_2, C_0, A, A_1, A_2, \tau \rangle$ where

- L_i is a finite set of time-stamped states for agent i (i.e., a combination (s, t) of a state and a positive integer),
- $C_0 \subseteq L_1^{t=0} \times L_2^{t=0}$ is a set of initial configurations drawn from the agents' states with time-stamp $t = 0$—we refer to $L_1 \times L_2$ as the set of possible system configurations,
- A is a finite set of actions,
- A_i is a function from L_i to 2^A that determines the actions that are physically possible for agent i (as a function of its state),
- τ is a (partial) transition function $\tau : L_1 \times L_2 \times A \times A \to L_1 \times L_2$ such that if a state l in a configuration c is mapped to l' in configuration c' under the function τ then the time-stamp associated with l and the time-stamp associated with l' are consecutive integers (i.e., a joint action will lead an agent from a state with time-stamp t to a state with time-stamp $t + 1$).

In this definition, two states with different time-stamps, but otherwise identical, will appear as two different states. However, in general, we will assume a finite upper bound T on the number of steps necessary to reach one's goal. Therefore, for all practical purposes, the set of states L_i can be considered to be finite.

Definition 4.2. A plan for agent i is a total function from L_i to A, such that the action prescribed to agent i by the plan at any state $s \in L_i$ is in $A_i(s)$.

An execution of a plan $\mathcal{P}$ by agent i is a sequence $s_0, s_1, \ldots, s_k$ of states in L_i, where s_0 is the state of agent i in c_0, $c_0 \in C_0$, and where the s_i's are the states visited by agent i when it follows its plan and the other agent follows one of its possible plans. An execution of a joint plan (consisting of one plan for each agent) is the sequence of configurations reached by following it (by both agents respectively).

Definition 4.3. A liveness goal for agent i is associated with a subset of L_i.

Intuitively, a liveness goal will be fulfilled if one of the states in the corresponding set is reached (i.e., if the agent passes through this state).

Definition 4.4. A *safety* goal is associated with a subset of $L_1 \times L_2$.

A safety goal g_{safe} is fulfilled if the system reaches *only* configurations in g_{safe}.

A plan for agent i is said to *guarantee* a liveness goal s_{goal} if all of its executions include at least one state $s \in s_{goal}$ (not necessarily the same for each execution), and the length of the prefix up to this state in each execution is polynomially bounded (in the size of the system, that we take to be $|A| + \max_i |L_i|$). A system is said to guarantee a safety goal g_{safe} if there does not exist an execution of a joint plan in the system that includes configurations which are not in g_{safe}.

4.1. Social laws and minimality

A social law σ is a set of functions, one for each agent, that restrict the plans available to the agents. Formally,

Definition 4.5. Given a system S, a social law σ in S consists of functions $\langle A'_1, A'_2\rangle$, for agents 1 and 2 respectively, where A'_i is a function from L_i to 2^A that defines the subset of actions prohibited for agent i in each state ($A'_i(s) \subseteq A_i(s)$ for every agent i and state $s \in L_i$).

A social law σ and a system S induce a *social system* S_σ similar to S, where the A_i functions are altered based on the functions A'_i (i.e., in the state s only actions in $A_i(s) \setminus A'_i(s)$ are allowed).

Definition 4.6. A social law σ in S is useful if the system guarantees each safety goal (regardless of the law-abiding strategies chosen by the agents), and if, for every liveness goal s_{goal} of agent i, there exists a plan $\mathcal{P}$ in S_σ that guarantees s_{goal}.

Consider the set Σ_S of useful social laws for the system S. We define a partial order $\prec$ on Σ_S: given two social laws $\sigma_1 = \langle A_1^{\sigma_1}, A_2^{\sigma_1}\rangle$ and $\sigma_2 = \langle A_1^{\sigma_2}, A_2^{\sigma_2}\rangle$ in Σ_S, we say that $\sigma_1 \prec \sigma_2$ if $A_i^{\sigma_1}(s) \subseteq A_i^{\sigma_2}(s)$ for all i and all $s \in L_i$, with at least one strict inclusion for one s and i.

Definition 4.7. A minimal social law σ_i is a useful social law such that there is no useful social law σ_j in Σ_S, $\sigma_j \neq \sigma_i$, that satisfies $\sigma_j \prec \sigma_i$.

Notice that a social law may fail to exist. However, if a social law exists, a minimal social law will exist as well.

4.2. Automatic synthesis of minimal social laws

Roughly speaking, the algorithm we have in mind when searching for minimal social laws starts from a useful social law and decrements the set of constraints. Hence, the basic question we are faced with is the following. Given a system, an appropriate useful social law, and a pair (s, a) of a state s and an action a, where a is prohibited in state s for agent i (by the given law), can we allow i to take action a in s and still remain with a useful social law? The answer to this question reveals an interesting connection between problems of planning with incomplete information and the design of minimal social laws. The following theorem shows that this question, which addresses a most basic problem in the construction of minimal social laws, is NP-hard.

Theorem 4.1. *Given a system S, and a useful social law σ that prohibits action a in state s of an agent i, deciding whether by allowing a in s we get a useful social law, is* NP*-hard.*

Proof. We prove the result by a reduction from Planning while Learning [52]. In Planning while Learning (PWL), we are given two automata, one for the agent and one for what

we call the environment. The agent has a set Q of observable states, a set of possible actions $\mathcal{A}$, an initial state $q_0 \in Q$ and an actual transition function. The agent does not know the actual transition function of the environment but knows that it is one of s possible behaviors. By performing actions, the agent can gain knowledge about the actual behavior of the environment. The problem that we face is to design a plan for the agent to obtain his goal for any environment behavior (taking into account the fact that by acting and getting feedback, the agent can learn about the structure of the environment). Suppose we are given a Planning while Learning system S which consists of two automata, one for the agent and one for the environment (see [52]). We now construct in polynomial time a two-agent system. In this two-agent system, the set of states for agent 1 includes a single initial state s_0^1 in which the agent can choose only one action a. The second agent starts in state s_0^2 where two actions x and d are available. The transition function is as follows: If agent 2 takes action x in s_0^2 then it reaches a state $goal_2$ where her goal is achieved, irrespective of the action chosen by her companion. Agent 1 will also reach his goal $goal_1$ by doing a in s_0^1, provided agent 2 did not take action d concurrently. In a case the joint action is (a, d), agent 1 moves to a state q_0 from which the specification of the agent's automaton (states and transition function) replicate the agent's automaton of the Planning while Learning system S. Similarly, agent 2, by taking d, reaches a state from which she chooses among s actions corresponding to the s behaviors of the environment in the Planning while Learning system S, all of them leading indistinctly to her goal $goal_2$. Initially, a useful social law forbids action d, and leaves all other actions legal. We show that social law which corresponds to no restrictions for both agents (i.e., extending the previous law by allowing d at s_0^2) is useful if and only if there exists a satisfactory plan for the agent in the PWL system. Clearly, if there exists a satisfactory plan for the agent (say P_s), then the plan where agent 1, in the derived two-agent system, chooses a and then, if agent 2 chose d conforms to P_s, leads agent 1 to his goal no matter how agent 2 behaves (if agent 2 chose a, agent 1 reaches his goal immediately as specified above). Therefore, in this case, adding d keeps the system useful. Likewise, if adding d keeps the system useful, there must exist a plan for agent 1 that guarantees reaching his goal (from the definition of usefulness). And then, the sub-plan of this plan which starts in q_0, represents a satisfactory plan for the PWL system. □

5. The minimal social law algorithm (MSLA)

The above result leads us to consider a special class of systems. In this section we consider the case where an agent's basic goal is to follow a predefined plan $\mathcal{P}_i$. Nevertheless, this goal or the agents' capabilities may change and therefore we do not wish to restrict the agents by requesting them to follow this and only this plan. We show that an efficient incremental algorithm for the computation of minimal social laws exists for this class of systems.

Consider a two-agent system S with a single initial configuration (i.e., $C_0 = \{c_0\}$). Consider a pair of plans $\mathcal{P}_1$ and $\mathcal{P}_2$, where $\mathcal{P}_i$ is a plan for agent i. Let t_i be a bound on the number of steps of $\mathcal{P}_i$. Let $(s_0^i, s_1^i, \ldots, s_{t_i}^i)$ be the execution of the joint plan $(\mathcal{P}_1, \mathcal{P}_2)$ projected on the states of agent i. We will associate this execution with the goal of

agent i. In the AGVs example, this sequence can be associated with a sequence of stations $l_0, l_1, l_2, \ldots$, where l_{i+1} and l_i are neighboring stations. Basically, this kind of goals is quite typical in systems where there are given protocols to follow, and where the role of a social system is to allow maximal freedom by relaxing unnecessary constraints. Notice that in this setting, agents are required to fulfill only their liveness goal and there is no safety goal. However a variant of safety is implicit in our definition of liveness goal, since an agent not following his state sequence will never be able to fulfill his liveness goal. Yet this is not properly speaking a safety goal. We do not wish to rule out the possibility that the agent might switch to a different goal, and thus we should let an agent depart from his goal if he wants to.

5.1. The MSLA algorithm

(1) Let s_k^i denote the kth state to be visited in the execution that corresponds to the original goal. Let s_0^i denote the initial state of agent i. Let $(a_1^i, \ldots, a_{t_i}^i)$ be the sequence of actions executed by agent i in the corresponding plan (P_i).
(2) Initialization step: let $k = 0$, and let $A'_i(s) = \emptyset$ for all i and s (intuitively, $A'_i(s)$ represents the set of forbidden actions at s).
(3) Let B_k^j be the set of reachable states at step k for agent j when agent $3 - j$ follows his (original/basic) plan. Initially, B_0^1 contains the initial state of agent 1 in c_0, B_0^2 contains the initial state of agent 2 in c_0, and all the other B_k^j's are empty sets.
(4) For each state $s \in B_k^1$ and for each action $a \in A_1(s) \setminus A'_1(s)$, if $\tau(s, s_k^2, a, a_k^2) = (s^1, s_{k+1}^2)$, where $s^1 \in L_1$, then add s^1 to B_{k+1}^1. Otherwise, add a to $A'_1(s)$.
(5) Increment k by one. While $k < t_i$, go to (4).
(6) Execute steps (4)–(5) again, switching the indexes for the agents.
(7) Output the social law whose components are the functions A'_1, A'_2.

Proposition 5.1. *Given a system S, and a goal that is a projection of a given joint plan, MSLA outputs a minimal social law for the system S in polynomial time.*

Proof. We first prove that the algorithm runs in polynomial time. The initialization takes $O(\max(L_i))$ iterations. At the very most, all the states in L_i are checked once in step (4) (states include a time-stamp). Given a state and an action, we assume that there exists a data structure (for instance a table) that maps the transition function τ (i.e., outputs the next state for the agents) in at most $O(\max(|L_i|).|A|)$ time units. Therefore, steps (4) and (5) take $O(\max(|L_i|) \cdot |A|)$ time. All in all, the algorithm is polynomial.

In order to show that MSLA outputs a minimal social law, we must first prove that at the end of the algorithm, the social law σ that the algorithm outputs is a useful law for the system S. Each agent i has a strategy that obtains the goal. Indeed, the original plan $\mathcal{P}_i$ is always preserved along the algorithm as no other agent's action that could interfere with agent i's plan (and make it depart from the original plans) is allowed (step (4) rejects any strategy of agent j which "threatens" agent i's plan). As there is no safety goal, σ is useful. In order to prove that the law is minimal, we suppose, for a contradiction, that there is an action $a \in A'_i(s)$ (for some s) that could be removed from $A'_i(s)$, such that the resulting

social law σ' would still be useful. However, if such an action exists, it must have been added at step (4) of the algorithm. Therefore it means that taking this action will make the other agent depart from her original plan (otherwise it would not have been added then). By making this action legal, we have introduced the possibility that the other agent will not be able to reach her goal anymore, since the first agent can force her out of her goal by choosing his new action a at s. Therefore σ' is not useful. This contradicts the assumption above and thus σ is minimal. □

5.2. Solving the AGV case study

The AGV case study presented in Section 3.1 can be represented and solved within our model. We now describe a sketch of how this can be done. Each robot has a finite set of states L_i that is a product of two components. Each state of one of the components refers to a station the AGV may be in, and each state of the other component refers to a station it may be in and observations it may have (i.e., in this second component a state refers both to a station and to the local observations made). There is an initial state where the agent selects whether it will observe only its station or the neighboring coordinates as well (i.e., a decision about the component of states to be visited). There is also a distinguished state that denotes collision. The goal is to follow a particular path along the ring without (the need to) observing the neighborhood, and without colliding. This can be easily defined by a path of states of the first component. The transition function will capture movements in the ring topology and potential collisions. The MSLA algorithm can be used now in order to build a minimal social law that is similar to the one presented in Section 3.1. The initial goal (according to which the minimal social law in MSLA is constructed) captures in fact the useful social law based on which the corresponding minimal social law in Section 3.1 is built.

At this point, we are armed with a new tool (minimality) to choose a 'right' social law (at least as long as we are concerned with flexibility), have tried it out on two domains of interest, and have examined computational issues related to it. Is this criterion ultimate? Do other criteria exist and what will their interpretation be? How do they compare with minimality? To answer these questions, we shall consider an alternative to minimality, a task that we undertake in the next section. We will consider a criterion *a priori* different from minimality, that we call simplicity. We shall draw connections between the two concepts in Section 8.

6. Simplicity

6.1. Introduction

In the previous sections, we introduced guidelines for the design of 'good' social laws. Usefulness, as the cornerstone of the approach, was a necessary requirement but was in many cases too poor a criterion to meaningfully lead the designer to a clear-cut choice among equally-useful social laws. We therefore proposed a criterion that we called minimality which attempts to find less constrained useful social laws.

Minimality sought to minimize constraints and exploited the full capabilities of the agents. Our intent in this section is to suggest an alternative approach to minimality. We consider a different concept, centered around the agent and its possible limitations. In this approach, we attempt to relax the need for the agent to rely heavily on his sensoring capabilities (or to possess such capabilities), in order for it to be able to comply with the law. Our purpose is to focus on simple social laws, i.e., useful social laws which are applicable to a variety of agents with a range of sensors (as well as to agents without these sensors). The idea behind simplicity is that some agents might be able to follow only simple laws, since a non-simple law could rely on capabilities not available to them. Moreover, simplicity reduces the sensitivity of a given system to modifications of the agent's capabilities. Roughly, if a law makes use of the whole range of an agent's sensoring capabilities, a slight change in these capabilities (for instance, as a consequence of faulty sensors) will render the law inadequate and require re-design of the system. Notice that although we view this improved independence of the law from the agent's sensoring capabilities as the main focus of our study of simplicity, *there will typically exist other important reasons for looking at simple laws*: learning the law is likely to be faster, and the representation of the law is likely to be more succinct.

The idea of simplicity is not new and can be found in research on bounded rational players in game theory. Rubinstein [51] proposes a representation of strategies as finite-state machines. In this representation, a strategy is modeled as an automaton in which each state incorporates also the action the strategy instructs the player to take at a specific situation, and transitions are driven by the other players' actions. Since a social law can be seen as a set of strategies, a similar representation could be used to model our concept of simplicity. Our definition of the concept of simplicity can be considered as a generalization of the idea that strategies are simple if they can be represented by a small automaton. In the context of social laws, reduced sensory capabilities will cause certain states to become indistinguishable; when states are indistinguishable, the constraints to be obeyed in these states should be identical.

In order to discuss the advantages associated with simplicity, we need a precise definition of what a simple law is, which we provide in this section. We also dwell on the importance of simplicity and argue that using the concept as a criterion to distinguish between candidate useful social laws is a plausible and recommended approach.

6.2. Simple social laws

> Everything should be made as simple as possible, but not simpler.
> —*Albert Einstein, cited in* [36]

6.2.1. A motivating example

Imagine you are driving on a bi-directional two-lane country road, with one lane separated from the other by a dotted line. Traffic rules require that you keep driving in the right lane and use the left lane only to overtake another vehicle. Of course, you shall not attempt to overtake if a third vehicle is moving toward you in the opposite lane. Such laws have been accepted in many countries since they guarantee that no accidents will occur as a result of collisions when overtaking. Note however that they rely on your ability

to identify situations where a vehicle is coming in the opposite direction. Is this ability absolute? How much can you trust it when you are driving at night, with no city lights or with unreliable head-lamps? In such cases it would probably be better to follow more restricted laws, e.g., by forbidding overtaking.

The preceding example underscores an important point about simplicity. Restricting actions is only one way to attain a lesser degree of dependency on sensors. Allowing unrestricted freedom could sometimes be an alternative way to make sensor dependency less critical. Thus, our definition should be general enough to encompass both stringent and less stringent laws.

6.2.2. Formal definitions

We now formalize the notion of simplicity. We use the computational model defined in Section 4. For convenience, we require also that the set of actions prescribed by the law at a state s_i of agent i includes at least one action. It allows us to ensure that an action be performed at each state (but it could be the null action or "do nothing"), and therefore provides us with a convenient way to express necessity. Note also that in the computational model, a law defines, at a given state, a subset of the physically possible actions at this state. In the literature, this property is called physical adequacy.

We would now like to provide a measure of the complexity of a useful social law *sl* in a given system S, and to compare social laws according to this measure. Clearly there are several ways to do so and we choose a general definition that captures the motivation presented above.

Definition 6.1. Consider a two-agent system $S = \langle L_1, L_2, A, A_1, A_2, \tau \rangle$ and a useful law *sl* for S such that $sl = \{A'_1, A'_2\}$. The partition P_i of the state space S_i of agent i is the set of the equivalence classes over S_i under the following equivalence relation R: for two states s_1, s_2 in S_i, $R(s_1, s_2)$ iff $A_i(s_1) \setminus A'_i(s_1) = A_i(s_2) \setminus A'_i(s_2)$

It is easy to check that R is indeed a reflexive, symmetric and transitive relation. Intuitively, two states s_1 and s_2 will be in the same element of the partition P if and only if the set of actions allowed by *sl* in s_1 is exactly the same as the set of actions allowed at s_2. Once we view a social law as defining a partition over the state space of the agent, a measure of complexity can be related to the complexity of the partition it defines. This connection is drawn in the following definition.

Definition 6.2. Given two useful social laws l and l' for a (two-agent) system S, each one with corresponding partitions (for each agent respectively) (P, Π) and (P', Π'), l' is simpler than l if for every element $P_k \in P$ and $\Pi_r \in \Pi$, there exist k' and r' such that $P_k \subseteq P'_{k'}$ (respectively $\Pi_r \subseteq \Pi'_{r'}$), with strict inclusion for at least one k (or r).

In order to illustrate our definition, let us consider a situation modeled as a multi-stage two-player game where each player plays either a full cooperation game (whose matrix payoff is given in Fig. 1), or a prisoner's dilemma game (Fig. 2), alternatively. We assume that the specification requests each player to always achieve the best payoff he could have achieved given the strategy played by the other player (in other words, to play a Nash

	C	D
C	a,a	0,0
D	0,0	a,a

Fig. 1. A full cooperation game, with $a > 0$.

	C	D
C	b,b	d,a
D	a,d	c,c

Fig. 2. Prisoner's dilemma with $a > b > c > d$.

equilibrium). Let us examine the law "play D when in the prisoner's dilemma stage and C otherwise". This law, that we will call σ, is useful. It splits the state space of each agent into two subsets, associated with playing D and playing C. Can σ be simplified? If we require our agent to always play D, the resulting law σ' will be simpler than σ. Indeed, imagine that a player loses the capability to discern between the two different stages of the game or that we, as designers, are restricted as to the number of actions we can implement into our agents, then σ' offers an attractive alternative to σ. Note that in our search for a simple law we again only consider useful social laws, as non-useful laws have little interest whatsoever for the designer, even if they are simple. Therefore, the task of finding a good social law with which the designer is charged is a double performance: restrict as less as possible the freedom granted to the agents in order to benefit them all, and design social laws that are simple enough to be followed by even primitive agents.

7. A computational study of simple social laws

In this section we consider computational issues related to simplicity.

7.1. Designing simple laws is not simple

We focus on the simplest kind of simple laws, i.e., social laws where each agents' partition consists of a single element. Note that the law need not be identical to each agent, provided it defines a single element partition for each one of them. We show that deciding whether such a law exists is an NP-hard problem.

Theorem 7.1. *Given a* (*two-agent*) *system S with a single initial configuration c_0 and where each agent has a single* (*liveness*) *goal g_i, deciding whether there exists in S a useful social law with single-element partitions in the agents' state space is* NP*-hard.*

Proof. The reduction is from 3-SAT (see [22]). Assume that we are given an instance of 3-SAT with k clauses and let n be the number of different primitive propositions in the formula. We consider the two-agent system where the agents are represented by two automata A_1 (Fig. 3) and A_2 (Fig. 4) as follows. A_1 has $k+1$ states among which a single initial state s_1^1 is associated with the first clause in the 3-SAT formula, a sequence of $k-1$ intermediate states $s_2^1, \ldots, s_k^1$, each associated with one of the $k-1$ other clauses of the 3-SAT formula, respectively, and one goal state s_{k+1}^1. A_2 has $n+1$ states with

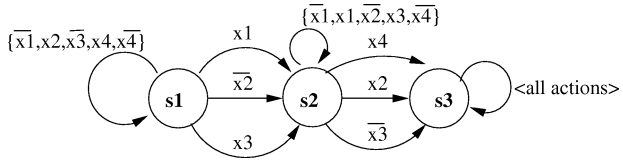

Fig. 3. The automaton A_1 corresponding to the 3-SAT formula: $(x_1 \vee \overline{x_2} \vee x_3) \wedge (x_4 \vee x_2 \vee \overline{x_3})$. There are 8 actions, and three states.

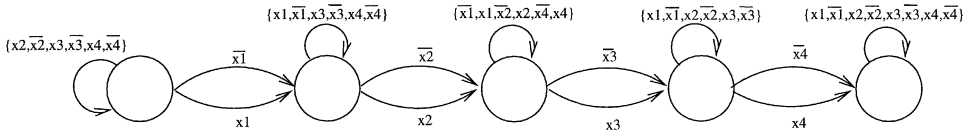

Fig. 4. The automaton A_2 corresponding to the 3-SAT formula: $(x_1 \vee \overline{x_2} \vee x_3) \wedge (x_4 \vee x_2 \vee \overline{x_3})$. There are 8 actions, and five states.

initial state s_1^2, a sequence of $n-1$ intermediate states $s_2^2, \ldots, s_n^2$, and one goal state s_{n+1}^2. (Notice that the number of non-terminal states in A_2 corresponds to the number of primitive propositions in the 3-SAT formula.) We define the set of possible actions A at any state of the agents to be $\{a_1, \ldots, a_{2n}\}$, each action corresponding to a literal or its negation. The transition functions are as follows: whenever agent 2 is in s_i^2 ($i \leqslant n$), performing the action corresponding to literal l_i or to its negation moves him to s_{i+1}^2. Otherwise, he remains in the same state s_i^2. At each state, all the actions are therefore physically possible, but only two are moving the agent one state ahead. As for agent 1, if the agent performs at a state s_i^1 an action corresponding to l, while the other agent performs the action corresponding to $\overline{l}$, A_1 moves to its special state *bad* from which it cannot escape (any action leads him back to *bad*). Otherwise, if the agent performs at s_i an action corresponding to one of the propositions in the associated clause negation of the it moves to the next state s_{i+1}. In all other cases, it remains in s_i. Each agent's goal is to reach his last state (s_{k+1} for agent 1 and s_{n+1} for agent 2). We now claim that the 3-SAT formula has a satisfying assignment if and only if the system has a simple social law with one element in each agent's partition. Assume that α is a satisfying assignment for the formula. We define the social law Σ (identical for both agents) to allow all the actions where l is true under α. The set of actions is the same for every state of the agent (i.e., the partition of the law has only one class). Since α is a satisfying assignment, it follows that for each clause there is at least one literal which is true. Therefore agent 1 can reach the goal state by performing the corresponding action of the appropriate literal at each state. Likewise, either one literal or its complement will be true and therefore agent 2 can reach his goal as well. Moreover, no literal l and its complement will both be true (α is a satisfying assignment) and thus agent 1 will never reach *bad* (as it would mean performing the action associated with l while the other agent performs the action associated with $\overline{l}$ contradicting the fact that α is a satisfying assignment). It follows that the social law is useful and simple with partitions composed of a single class. It remains to show that if a simple social law (with one element) exists, then α is satisfiable. Note that the simple social law need not specify the same action functions

for both agents. Let Σ be such a law. At each clause corresponding to a state s^i of agent 1, assign the value true to literals associated with the actions authorized by the law at s^i. As Σ is useful, there is, in each state, an action a which leads to the next state and therefore, Σ defines a partial assignment α to the literals of the formula. Suppose for a contradiction that in this assignment, a literal l and its negation both get a true value. However, Σ must allow agent 2 to take at least one of the actions corresponding to l_i or $\overline{l_i}$, for all $i \leqslant n$, in order for the agent to be able to reach his goal. Therefore agent 2 might perform the action associated with l whereas agent 1 might take the action associated with $\overline{l}$ at the same time (recall that the same set of actions is allowed at each state of an agent since the law defines one-state partitions). In this case, Σ would not be useful, and thus, α must assign a true value to either a literal l_i or its negation but not both. α can be completed by assigning a true value to literals which did not receive a true value, and neither did their negation. The resulting assignment is a satisfying assignment for the 3-SAT formula. □

The reader should notice that in the proof of the above theorem we did not explicitly incorporate time-stamps in the states of the automata. The reason for doing so is ease of exposition since one can easily derive automata with time-stamps from our construction without challenging the correctness of our analysis. In this process, any action leaving an agent in his current state would be modeled in the time-stamped automaton as a transition between two states which differ only by their time-stamp. Overall, if the original automaton has m states and the bound on the plan is k ($k = \mathrm{O}(m)$), then the extended automaton (incorporating time-stamps in the state) will have $m(k-m)$ states, i.e., $\mathrm{O}(m^2)$ states. The construction remains therefore polynomial in the length of the formula.

7.2. Homogeneous simple social laws

What can be said if we require the social law to define identical partitions in the state space of the agents? Clearly, such 'homogeneous' simple social laws are a subset of the set of available simple social laws in the system. However, this additional constraint does not make the design of one-state simple social laws computationally easier.

Theorem 7.2. *Given a* (*two-agent*) *system S with a single initial configuration where each agent has a single goal, deciding whether there exists in S a useful social law* identical to all the agents, *with a single element in their partition, is* NP-*hard.*

Proof. The proof resembles the proof given in the preceding theorem and uses a slightly different reduction from 3-SAT. Assume that we are given a 3-SAT formula with k clauses and n primitive propositions. We consider the following two-agent system: the agents are modeled as identical automata A_1 as defined in the proof of Theorem 7.1 (see Fig. 5 for an example). Each has a single initial state s_1 associated with the first clause of the formula, a sequence of $k-1$ intermediate states, each associated with one of the $k-1$ other clauses of the 3-SAT formula, respectively, and one goal state s_{k+1}. There is also a special state *bad* whose purpose will be explained later. The set of actions is identical at each state of both agents. It includes $2n$ actions, each corresponding to one literal (or its negation) of the 3-SAT formula. The transition function is as follows: if an agent performs at a state s_i

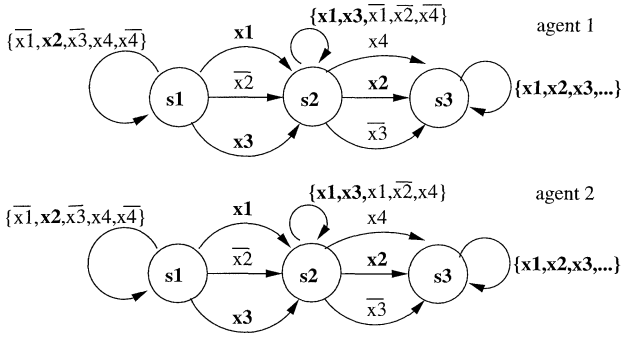

Fig. 5. A two-agent system where each agent is represented as the automaton of Fig. 3. The system has a simple homogeneous law, with one class in each agent's partition. Actions allowed by the law at each state are shown in bold.

($i \leqslant n$) an action corresponding to l and the other agent takes simultaneously the action corresponding to $\overline{l}$, both move to their state *bad* from which they cannot escape (any action returns them to *bad*). If he performs an action which corresponds to one of the literals of the clause with which s_i is associated, he moves to the next state s_{i+1}. Otherwise, he moves back to s_i. At state s_{k+1}, any action moves the agent back to s_{k+1}. We now claim that the 3-SAT formula has a satisfying assignment if and only if the system has a (homogeneous) simple social law with one element in the agents' partition. Assume that α is a satisfying assignment. We define the social law Σ to allow only actions where l is true under α. This set of actions is identical across the states of the agents. Since α is a satisfying assignment, it follows that for each clause, there is at least one literal which is true. Thus, the agent can reach his goal by performing the action corresponding to this literal at each state. Also, no literal and its complement will both be true, and therefore no agent will ever reach his *bad* state. Hence, Σ is useful. Now, assume that Σ is a homogeneous simple social law for our system. Since it has only one element in the agents' partition and it is useful, the same set of actions is legally possible at each state. Define the (partial) assignment α to give a true value to the literals of the formula whose corresponding actions in the system are allowed by the law. Clearly, at least one literal in each clause receives a true value. Also, since the law defines identical action functions to both agents, should two conflicting actions (i.e., actions associated with a literal and its negation) be allowed by Σ, they would have to be allowed at any state, and to both agents. As a consequence, an agent could perform one of these actions, while the other would perform the second, and both would move to *bad*, contradicting the usefulness of the law. Therefore in α no literal and its negation receive a true value. α can be completed by assigning a true value to literals which did not receive a true value, and neither did their negation. The resulting assignment is a satisfying assignment for the 3-SAT formula. □

7.3. Quasi-modular systems

We now present a special class of systems, for which designing simple social laws with single-element partitions is efficient. In the following definition, τ_i refers to the ith

component of the image of a given state vector and action vector by function τ (in a two-agent system, this vector has only 2 components).

Definition 7.1. Let be given a two-agent system and let $s \in L_i, a \in A_i(s), s' \in L_{3-i}$, and $a' \in A_{3-i}(s')$. A quasi-modular system is a two-agent system S with an initial configuration c_0 and a single liveness goal g_i for each agent, such that:

For $i = 1$: If $\tau_i(s, s', a, a')$ is defined and $\tau_i(s, s', a, a') \neq bad$ then for all $s'' \in L_{3-i}$, and $a'' \in A_{3-i}(s'')$ for which $\tau(s, s'', a, a'')$ is defined, one of the following is true:
- $\tau_i(s, s'', a, a'') = \tau_i(s, s', a, a')$.
- $\tau_i(s, s'', a, a'') = bad$ (where *bad* is a designated state).

For $i = 2$: If $\tau_i(s, s', a, a')$ is defined and $\tau_i(s, s', a, a') \neq bad$ then for all $s'' \in L_{3-i}$, and $a'' \in A_{3-i}(s'')$ for which $\tau(s'', s', a'', a')$ is defined, one of the following is true:
- $\tau_i(s'', s', a'', a') = \tau_i(s, s', a, a')$.
- $\tau_i(s'', s', a'', a') = bad$ (where *bad* is a designated state).

In other words, if τ is defined at the appropriate states, then $\tau(s, s'', a, a'')$ leads agent 1 either to the same state as $\tau(s, s', a, a')$ would, or to a *bad* state (and similarly for agent 2). A *bad* state will typically correspond to a dead-end state from which reaching the goal is impossible. A quasi-modular system is therefore a two-agent system where transitions that do not lead to a *bad* state are fully determined by the action chosen by the agent, irrespective of the actions of the other agents. This characterizes systems where interactions may prevent agents from achieving their goals, but (otherwise) do not modify the outcome of their actions.

Definition 7.2. A *moderate* quasi-modular system is a quasi-modular system that satisfies $|\bigcap_{s_i} A(s_i)| = \mathrm{O}(\log(\max_i |L_i|))$, for all i.

We can now state the following theorem.

Theorem 7.3. *Given a moderate quasi-modular system S, finding a simple social law with one element in the associated partition for S (if exists, and otherwise announce that no such law exists) is polynomial.*

Proof. We consider the homogeneous case. The non-homogeneous case is treated similarly. The proof is constructive: we show that we can enumerate the candidate social laws and check their usefulness in polynomial time. There are only polynomially many such laws as we have a logarithmic number of actions that can possibly constitute the set of actions allowed at any state (those in $\bigcap_{s_i} A(s_i)$) for each agent. It remains to show that the verification process can be done in polynomial time. Given a candidate social law, we simulate for each initial configuration (there are polynomially many) the possible paths that an agent can follow when obeying the given social law. The result of this simulation is a di-graph (it's a directed acyclic graph as states are time-stamped). In order to construct this graph, we can first ignore the interactions between the agents and get a graph G without

any transition to *bad*. We then extend at each state the digraph with transitions to *bad* if any, based on the set of states that the agents can be in at the given time (as reflected by G). As the system is quasi-modular, there will not be any other interactions between the agents. In particular, this means that we do not need to consider the actions the second agent took at earlier moves. The construction of the graph is bounded by $(\max_i |L_i|)^2 \cdot |A|^2$. We first mark each agent's goal states (included in g_i) as *good*. Then we proceed backwards in the graph by labeling a node as *good* if and only if there exists an action leading from the node to a state marked as *good* and there is no possibility that this action will take the agent to a *bad* state. Provided that we keep appropriate data structures (for each state, a list of actions that can lead to *bad*, and a global list of states that have been visited by the backwards induction procedure), this will take $O(|L_i|^2 \cdot |A|^2)$. We stop as soon as all the initial states have been marked as good, or all the states have been visited, and turn to the other agent for which we check the law similarly. If all the initial states of both agents have been marked as good, the social law is useful as there is a plan that guarantees achievement of the goal for each agent (by following the actions that have been used in labeling the different states as *good*). □

8. Simplicity and minimality

In this section we study the relationship between simplicity and minimality. Are simple social laws necessarily minimal? Does minimality entail simplicity? What can be said about systems in which minimal and simple social laws are related? These are some of the questions that we address here.

8.1. Connections between minimal and simple social laws

8.1.1. Minimality versus simplicity

The first question that we consider is whether minimality and simplicity can be inferred one from the other. Our analysis will use the computational model of the previous section.

Proposition 8.1. *Simple social laws are not necessarily minimal.*

Proof. Consider Traffic Law 1 for the AGVs case study (the ring environment) presented in Section 3. We showed at the end of Section 3 how to express this case study in terms suitable for analysis in our computational model. In this model, Traffic Law 1 is simple but not minimal. □

Proposition 8.2. *Minimal social laws are not necessarily simple.*

Proof. Traffic Law 2 in the ring environment is minimal but not simple. Indeed, it is possible to obtain from Traffic Law 2 (whose partition consists of three classes of states) a law with less classes. This is achieved by combining all the classes together and prescribing the action "clockwise" at all state. The resulting law is Traffic Law 1 which was proven to be useful. □

Consequently, we can find a minimal social law for the ring environment, which is not simple, and a simple social law, which is not minimal. This raises the question of whether, given a domain, there always exists a minimal and simple social law.

Proposition 8.3. *A simple and minimal social law does not necessarily exist.*

Proof. Again, the counter-example is the ring environment. To show that in this environment, no social law can be both simple and minimal, we first prove that any given social law with more than one element in its partition can be simplified. Given such a social law, we can unify the classes in the partition associated with the law (and obtain a single element partition) by requiring the agents to move "clockwise" at each state. Each class of the non-simple law is properly included in the single class of the resulting law. As this law is useful (it is Traffic Law 1), it is simple. Next we observe that no social law with one element in the associated partition is minimal. First, no social law which allows the *same* set of more than one action at all states (for example "stay in place or go clockwise") is useful as it cannot always prevent collisions. Each of the possible social laws with only *one* (same) action allowed at each state is not minimal, as we might relax the law by allowing an additional action whenever the neighboring stations of the current position of the robot are empty. For instance, the social law "go clockwise" can be extended if we allow a robot to stay immobile at all those states where no robot directly follows it. Therefore no simple social law is minimal. □

8.1.2. A simple and minimal social law

Although some domains do not lend themselves to both simplicity and minimality, the notions of simplicity and minimality are not orthogonal. In order to show this, we now present a well-known domain, the multi-robot grid environment [56], for which a useful and efficient social law can be proven to be both simple and minimal.

This domain is important as it has already been studied in the literature and an efficient social law has been proposed for it. The grid is frequently considered to be a simplification of a robot motion-planning environment, and it leaves enough degrees of freedom to make the analysis of social laws in general, and minimality/simplicity in particular interesting. We will be able to show that a useful and powerful social law presented in previous work on artificial social systems is both minimal and simple. The proof of this is nontrivial, and illustrates how the concepts of minimality and simplicity can be usefully applied to generate good social laws, and to verify properties of these laws.

A multi-robot grid system can be identified, e.g., with lanes in a supermarket or paths in a warehouse. Roughly speaking, it consists of a graph whose topology is a grid, on which several robots are moving. A more complete description of the domain appears in the proof of the following proposition.

Proposition 8.4. *A minimal and simple social law exists in the multi-robot grid domain.*

Proof. The structure of the environment is a square $n \times n$ grid (n is assumed to be even), with $m \leqslant 4(n-1)$ robots, located on coordinates of the grid. Each robot is fully aware of its current position on the grid, and has the capability to move to one of the stations in the

vicinity of its current position, or stay immobile. As time goes on, each robot moves around the grid and at discrete regular points in time, occupies one of the n^2 coordinates on the grid (in the sequel, "a station"). It then decides on the action to take as a function of its percepts (its current position) and the legally available actions at this state. A collision is identified with two robots on the same coordinate. The system is synchronous and movement to a neighbor coordinate takes one time unit. No perception of the surroundings is assumed (a robot cannot observe the state of the stations that surround it).

Row b will be the bottom row of the grid and row u the top row. Column l will be the second leftmost column of the grid. Initially the robots start operating from the $4(n-1)$ stations of the border of the grid, and each follows the same set of rules as follows. At each station of b, a robot has to move 'right' except at the rightmost station where the robots move 'up'. On l (provided the position is neither on u or b), the motion is 'up'. On u, robots move 'left', except when at the leftmost column where they move 'down'. At any of the other stations of the grid, both 'left' and 'up' are allowed. These rules will be referred to as Traffic Law 3, and are shown in Fig. 6.

Lemma 8.5. *Traffic Law* 3 *is useful.*

Proof. Let us refer to the coordinates of the leftmost column and the bottom row as the outer contour, and the rest of the coordinates as the "inside" of the grid. On the outer contour, a robot R could collide only at the top left corner as only one action is allowed at the states of the contour and robots have only one way to enter it, namely the top left corner. However colliding there would mean that both robots would have already collided

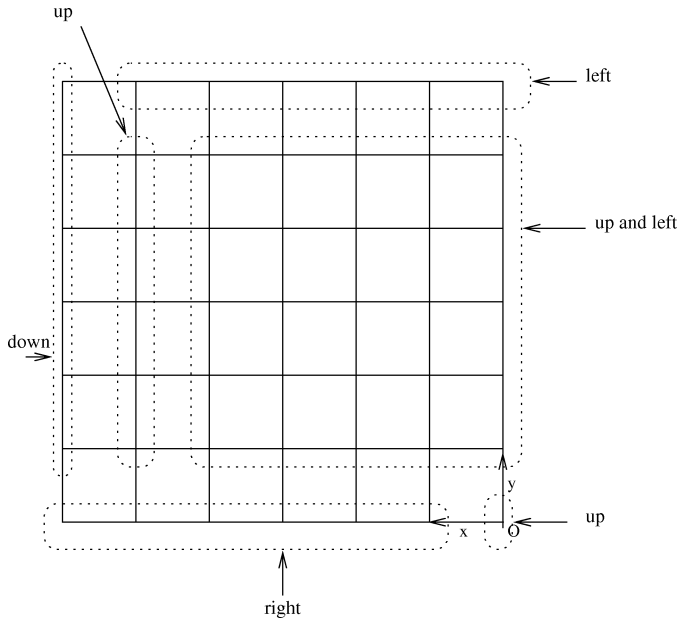

Fig. 6. The multi-robot grid environment and Traffic Law 3.

in a state of the inside of the grid, and thus would not have reached the top left corner. Let us define a system of axes as in Fig. 6, with origin at the lower right corner. The position of a robot in the inside of the grid is defined in this system by $x + y$. With the given social law, the position can only increase at each state by one, until R reaches the top left corner. Suppose that R collides with another robot after k steps, $k \leqslant 2n - 2$. Since both robots must have been previously at a state of the rightmost column, their position on this column would have been the same. This could only be if both started their motion at the lower right corner together, since at the starting configuration, no two robots occupy the same coordinate on the rightmost column. However, this would contradict the fact that R reached this corner without colliding. Therefore, we guaranteed that no collision would occur. Since R can choose an appropriate course of actions in the inside of the grid in order to reach any point of the grid, the law is useful. □

A variant of the law described in the above proof has been shown to be very useful and efficient in the context of traffic laws for mobile robots [55,56]. It is interesting to see that this existing law is both simple and minimal. This requires a somewhat elaborated proof that we now present. This gives us another support to the basic role that minimality and simplicity play in the design of desired social laws.

Lemma 8.6. *Traffic Law* 3 *is simple.*

Proof. Traffic Law 3 yields a partition with elements corresponding to the following subsets of actions: {up}, {down}, {right}, {left}, {up, left}. In order to draw a simpler social law, we consider the effect of unifying two classes around a single action as shown in Fig. 7. One should be aware that when considering a possible simplification for an agent i we cannot assume any assumption as to another agent's partition, since it might very well undergo a simplification process as well. However, whatever the law, it will always have to guarantee the specification. A case analysis (see Fig. 7) shows that it is not possible to find a partition that would replace the current partition and would keep the law useful.

(In all the following cases, refer to Fig. 6 where the law is drawn together with the state partition.)

1. Case 1: Let us suppose for a contradiction that a law exists where these states are grouped together under the action 'left', and the law is simpler. Consider the rightmost column. In order to reach it, action 'right' will have to be allowed at the states where {up, left} is currently required. But these states include the rightmost column where 'right' is physically impossible.
2. Case 2: In such a case, the coordinate corresponding to the second leftmost column, bottom row, is not guaranteed to be reachable anymore, contradicting the usefulness of the law.
3. Case 3: The top left corner of the grid is not guaranteed to be reachable anymore.
4. Case 4: The lower left corner of the grid is not guaranteed to be reachable anymore.
5. Case 5: The lower right corner is not guaranteed to be reachable anymore.
6. Case 6: The top right corner is not guaranteed to be reachable anymore.
7. Case 7: The top right corner is not guaranteed to be reachable anymore.

group	up	down	left	right
up with down	top left	lower right	top left	lower right
up with left	top	lower right	case 1	lower right
up with right	case 2	lower right	lower left	lower right
down with left	top	case 3	top left	top right
down with right	top left	bottom	top left	case 4
left with right	top	bottom	lower left	top right
up-left with right	case 5	bottom	lower left	rightmost
up-left with left	top right	case 6	case 7	top right
up-left with up	case 8	lower right	case 9	lower right
up-left with down	top left	case 10	top left	rightmost

Fig. 7. Simplifications of Traffic Law 3. The first column indicates the two sets of states to be combined. The action to be performed at the resulting common set appears in the next columns. In each cell of the table, we indicated the reason this unification is impossible. Whenever the reason stems from the fact that we would allow an action physically impossible at a state, we specify this state. Otherwise, we analyze the case independently in the text.

8. Case 8: Suppose for a contradiction that unifying around ‘up’ is possible. In this case, the action “down” at the leftmost column will have to be available since otherwise there will be no way to reach the coordinates of the bottom row. For the same reason, at the bottom row, ‘up’ will have to be made available (e.g., in order to reach the column in the middle of the grid). As a consequence, the lower left corner is not guaranteed to be reachable to other agents. Indeed, after the simplification, our robot could decide to move back and forth between this coordinate and the one above, preventing any other robot to reach the corner without colliding.
9. Case 9: The second row (from the bottom), rightmost column is not guaranteed to be reachable anymore.
10. Case 10: Let us consider the top right corner. This corner is not guaranteed to be reachable anymore if this simplification is made. Indeed, in order for the robot to reach the top row, it will need to move on the second leftmost column, as motion on other columns is ‘down’ only. However, once at the top row it will not be able to move ‘right’ (although this is the only way to reach the corner) since ‘right’ is not physically available at the rightmost corner, and all the states of the top row (except the leftmost corner) are in the same class. Therefore the law would not be useful.

To cover all possible simpler laws, we should also look at simplifications involving union of two classes around *more* than one single action. In most cases (all the cases of the rows in the table having only one “case x” in the row), such a simplification is not possible. Indeed, the reason for rejecting a simplification corresponding to a cell not marked as “case x” lies in the existence of a state where we would allow an action physically unavailable. Therefore, combining states by allowing this action, even with other actions, is not possible. The only two cases that we still need to consider are replacing the actions at the states corresponding to {up, left} and {left}

with {down, left} and the actions at the states corresponding to {up, left} and {up} with {up, left}. In the first case, the top right corner is not guaranteed to be reachable anymore, whereas in the second case a robot at the lower right corner can choose to move 'left', 'right', 'left', 'right' alternately (since 'right' has to be available at the other states of the bottom row in order for them to be reachable), precluding any other robot to reach the corner without colliding.

The case analysis exhaustively covers any potential set unification, and thus the law is simple. □

Lemma 3. *Traffic Law* 3 *is minimal.*

Fig. 8 shows a situation depicting a possible configuration, reachable from the initial configuration, where the actions at the leftmost column and the bottom row cannot be extended as a robot would collide. It is therefore only necessary to verify that we cannot allow a robot to go 'down' or 'right' when at the top row, 'right' at the second leftmost column, and either 'right' or 'down' at one of the remaining states.

1. Top row → right: If the robots are positioned one after the other on the top row, allowing to move 'right' will result in a collision (since moving 'left' is the normal behavior).
2. Top row → down: Let us suppose that we do allow the action 'down' at one of these states with coordinates (x, n) for robot i. Consider the following path for robot i: go around the border until reaching (x, n) and then go down. Now assume robot $i + 1$ follows robot i until reaching the second top row and then turns left. The two robots will collide.
3. Second leftmost column → right: Assume that robot $i + 1$ exactly follows robot i until they reach this column, at which point robot i takes a 'right' action and collides.
4. Remaining states → right, or down, or both: It can be readily seen that robots i and $i + 1$ could collide if we were to design a path for robot $i + 1$ such that both robots would find themselves at the same coordinate (the paths in Sections 2 and 3 above are examples of such situations).

Therefore the law is minimal. □

The proposition follows as a result of Lemma 3.

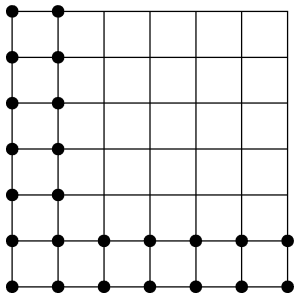

Fig. 8. A possible configuration of the multi-robot grid environment.

8.2. Policies and simple social laws

8.2.1. Are policies simple?

Since the concepts of minimality and simplicity seem to be loosely connected, we turn our attention to a special kind of social laws, policies. A (useful) policy is a (useful) social law that specifies for each state of each agent a single action. First, we show that useful policies are not necessarily simple. We then prove a strong connection between simplicity and minimality in the context of policies.

Proposition 8.7. *A useful policy is not necessarily simple.*

Proof. We consider the multi-robot grid environment with the following modifications. Initially, $m \leqslant n$ robots are positioned on the border of the grid such that two adjacent robots are separated by at least three stations (of the border). Our goal as designers, is to ensure that peaceful coexistence is guaranteed, i.e., no collisions occur and every point on the grid is reachable. We assume that a robot can move forward, backward, turn left or right (a 90 deg turn with no linear move), or stay immobile. The state of a robot is described by the following components:

- A binary variable to distinguish between odd and even rows.
- A variable to distinguish between the rightmost column, the leftmost column and the column adjacent to the leftmost.
- A variable specifying the location of the leftmost column with respect to the current position of the robot. It can take on one of the values 'left', 'right', 'front', 'back'. When on the leftmost column, this variable is defined as 'front' at the upper right corner, 'back' at the lower right corner and 'right' at all other places.

If all the agents follow a snake-like path (see Fig. 9) then they are guaranteed to reach any station on the grid in at most $O(n^2)$ time, without colliding. This path defines a function from the nodes on the grid to an action of A. Let us assume that the function associates one of the available actions 'forward', 'left' or 'right' indifferently at states which do not correspond to a situation encountered when following the snake-path on the grid (for example, states that are a combination of a coordinate and an orientation that does not appear in the snake-like path). As the path does not make use of 'backward' or 'stay immobile', we do not need to use more than three actions. Since the policy makes

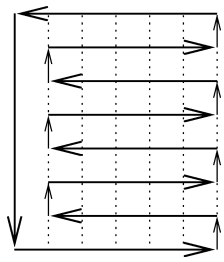

Fig. 9. A snake-like path on the grid.

use of the three other actions, the partition associated with the given social law has exactly three elements. However, note that we can model a left turn as three consecutive right turns: given the specification above, a robot has the capacity to distinguish between the stages of a left turn modelled as a right turn, i.e., before turning right, the intermediate states of the three consecutive right turns, and the following state where it has to go forward. The full specification appears in Fig. 10. Note that a left turn modeled as three right turns will take three steps instead of one but the original configuration above (where there is at least a three-station interval between any two robots) ensures that no collision happens. We can thus find an appropriate mapping between the state of the robot and the two actions 'right' or 'forward'. Therefore, we express the path function solely in terms of the state space as specified above and the two actions 'forward' and 'turn right'. The partition associated with this law has two elements and thus, the resulting policy is simpler. □

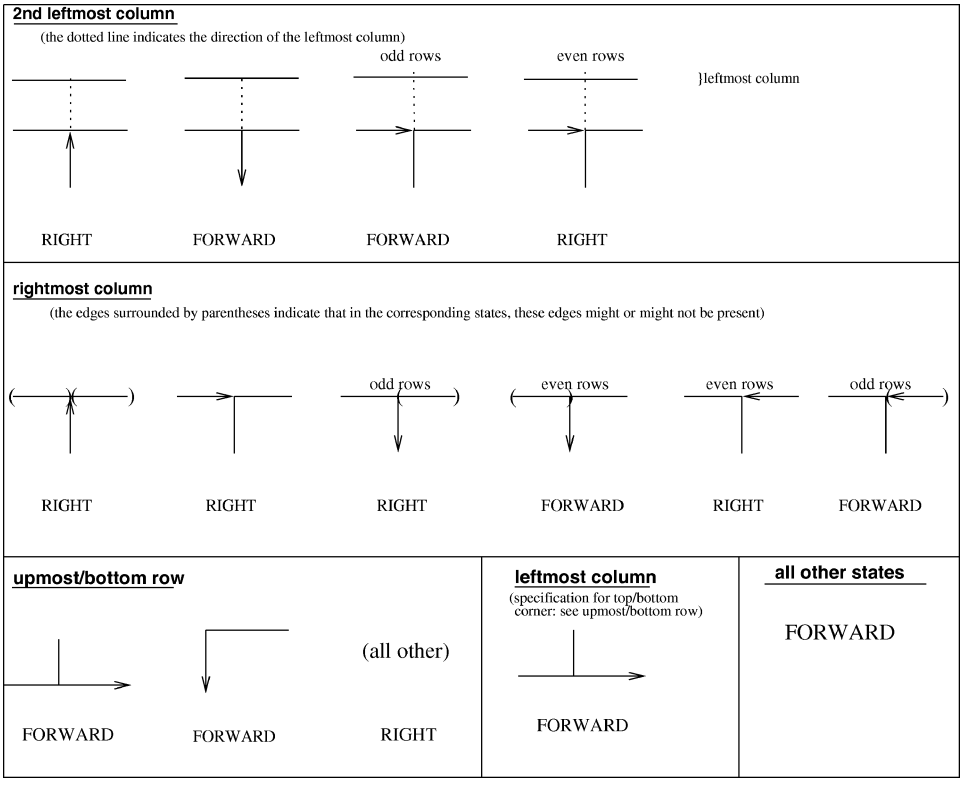

Fig. 10. A description of the law at the stations of the grid. The robot is positioned at a station represented as the intersection point between the edges. The edges stand for the incoming and outcoming tracks to and from the station. The orientation of the robot is given by the arrow. Whenever an edge is enclosed in parentheses, the corresponding states might or might not include this edge. The action prescribed by the law appears below the description of the state.

8.2.2. Weak simplicity

So far, we have found little correlation between simplicity and minimality, even in the realm of policies. However, if we restrict the process of simplification to what we call local simplification, strong connections can be stated.

Definition 8.8. Given a system S, consider a social law Σ and the partition $P = \{P_k^i\}_k$ of the states of agent i. Σ is locally simplifiable if there exist $P_{k_1}, \ldots, P_{k_l}$ such that:

(1) $P_{k_1}, \ldots, P_{k_l} \in P$.
(2) $P_{k_1}, \ldots, P_{k_l}$ are pairwise distinct.
(3) The following process yields a useful social law: modify the sets of actions allowed in $P_{k_1}, \ldots, P_{k_l}$ to make them all identical, and combine $P_{k_1}, \ldots, P_{k_l}$ in the resulting partition. The rest of the agents remain untouched.

The particularity of local simplification lies in the fact that the other agents remain with the same partition as in the original social law.

Local simplification allows us to consider a weaker notion of simplicity.

Definition 8.9. A social law *sl* is weakly simple if it is not locally simplifiable.

Proposition 8.10. *In quasi-modular systems, every minimal policy is weakly simple.*

Proof. Suppose we are given a minimal policy. Being a policy, it specifies a unique action at each state of the agent and thus there is no way to simplify the policy by forbidding the unique action prescribed by the law at some of the states (recall that our model requires leaving at least one action available at each state). Also, we cannot just add a new action at these states as the policy is minimal. Therefore, it remains to show that replacing the actions at some states by a single action (or group of actions), is not possible either. Assume for a contradiction that there is a state s for agent i where the law can be changed to include action b instead of action a at this state. It is clear that (see Fig. 11) s can only be a state on the path of states visited by agent i when following the original social law (we call this sequence of states $\mathcal{P}$). Otherwise, extending the original policy by adding b at s would not have altered the usefulness of the law ($\mathcal{P}$ remains untouched since the state is off-path) and it contradicts the minimality assumption. Thus s is a state on the path to the goal. Let s be the first state on $\mathcal{P}$ where the set of actions was changed by the simplification. As assumed, the social law remains useful after the simplification. Since a is not available at s anymore, the new path to the goal $\mathcal{P}'$ will rely on the agent taking (without loss of generality) action b at s. First suppose that $\mathcal{P}'$, after leaving s, includes only states which were "off-path" in the original social law. In this case, we could have added b at s contradicting the minimality of the law. Therefore, the path must rejoin $\mathcal{P}$ at some state x. Now look at the first state s' after the two paths have merged, where the set of actions have been changed by the simplification. The same reasoning shows that there must be a state "upward" in $\mathcal{P}$ where the two paths will rejoin again. Eventually, the two paths will merge after departing one from the other at some state w, and no states with new actions will be encountered anymore. But then, b could have been added at w in the original policy which contradicts minimality. Note that the fact that we are dealing with quasi-modular systems

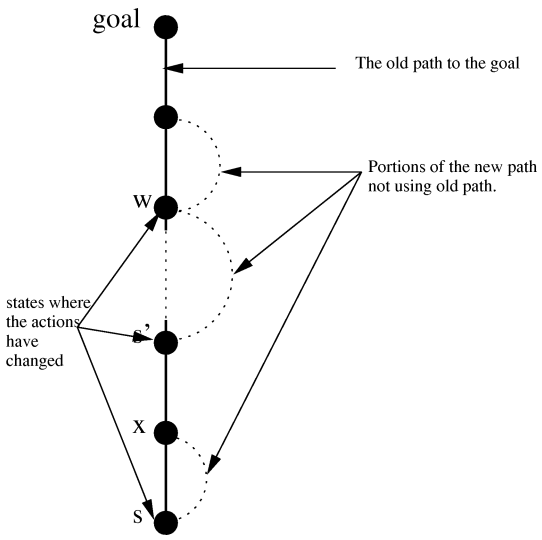

Fig. 11. The paths for agent i: $\mathcal{P}$ is shown in bold, the states s, x, w are shown on the path.

allows us to ignore the impact which the modification of the policy for one agent may have on other agents. At state x, the corresponding states of the other agents are exactly the same as when agent i followed his original policy since no agent will have been moved astray to *bad* (the law would not be useful) and the states are time-stamped. This observation is important as it justifies the fact that when deriving the contradiction, the states of the other agents were implicitly the same as when agent i followed his original policy. If it were not the case, it would be possible to find cases where a simpler policy exists without contradicting the fact that the original policy is minimal. □

9. Discussion and related work

In this section, we review previous work on which this research builds. We first discuss research that does not refer explicitly to artificial social systems, but addresses issues related to this paper. We then present the artificial systems approach, examine how this paper complements previous work on this topic and discuss implications that this research has on the design of artificial social laws.

Coordination is a central issue whenever multiple individuals with independent will, share a common environment. A possible way to coordinate such systems is to rely on central coordination (e.g., [32,58]). Another approach is to rely on fully decentralized solutions (e.g., [28,49,53]). Work that follows the latter approach has been characterized by game-theoretical models of negotiations. For example, Rosenschein and Genesereth [48] use such models to analyze how rational agents can coordinate their activities by striking deals among themselves. Kraus and Wilkenfeld [29] describe a situation of conflict that agents can solve by engaging in a negotiation process. The authors use classical bargaining

game theory [45] where the time at which a deal is reached is an integral part of the outcome of the negotiation. Rosenschein and Zlotkin [49] advocate the use of rules of encounter, i.e., rules stating the agent's behavior (and in particular, the structure of the negotiations) when the agent's activities interfere with those of another agent. Agents use communication to converge to a joint plan (a deal) whose payoff is higher than the one achieved when acting alone. The authors draw a relationship between categories of domains and negotiation mechanisms (e.g., they characterize a set of domains, Task Oriented Domains, where simple stable strategies lead to efficient outcomes). Other types of models and techniques are used in order to model joint commitments/intentions that are essential in order to follow desired activity in the corresponding decentralized multi-agent system [8,10,25,26]. Needless to say that both ideas of decentralized problem solving, as well as ideas with regard to the definition of joint commitments/intentions are relevant to work on artificial social systems and social laws.

Work in the areas of organization theory (see [34]), team theory [35], and organization roles (see for example [17]) is also of relevance. This work is especially concerned with the design of agents' roles and communication structures which enable a cooperative achievement of a common goal. The reader should notice that work on artificial social systems concentrates on a somewhat complementary issue: the off-line design and computation of laws which enable each agent to work individually and successfully toward its own goals during the on-line activity given that all the agents obey these laws. Additional related work includes the synthesis of multi-agent programs [46] and work on cooperative discrete event systems (DES) [47]. A main difference between these lines of research and the work in the framework of artificial social systems stems from the fact that the latter gives more structure for the design of multi-agent systems (first find a social law and then enable agents to tend to work individually) while concentrating on the fundamental problems of social design (we have to restrict the behavior of agents but not restrict them too much).

This paper uses a "Society" metaphor. The agents are treated as a society, for which minimal/simple useful social laws are designed. Society metaphors have been proposed in the AI literature also in contexts which differ from the artificial social systems setting. Minsky uses a society metaphor in his work on the society of mind [36]. The notion of *social choice* is an important element in, e.g., the work of Jon Doyle [15]. Finally, social metaphors appear also in the works of Fox [20], Kornfeld and Hewitt [27], Malone [34], and Simon [57] concerning organization theory. In this paper we treat the notion of an artificial social system in a relatively narrow sense, and with a particular point of view in mind. We wish to develop a theory to support the design of multi-agent environments, and to assist the designer by supplying tools for the evaluation of social laws.

In spite of the generality of our work and of the artificial social systems approach it is based on, we wish to emphasize that this work in no way diminishes the crucial importance of mechanisms for interaction and communication among agents, nor the significance of the study of effective representations for agents. Some of the work developed in LIFIA [4,14] provides considerable progress in these complementary directions. The discussion of various ways of modeling agents is of significant importance to the coordination between agents. Further study of artificial social systems may need to take various representation levels into account while addressing the construction of useful social laws. In addition, our

work does not take into account the incentives agents have for cooperation. Some of the work in CNR [7,11] has been concerned with issues such as defining goal adoption as a basic form of cooperation and the effect of social power on cooperation among agents. Although some of our technical machinery may enable to express concepts such as goal adoption, our emphasis is on the design of artificial systems where agents are assumed to conform to every law prescribed by the system's designer.

The law-based approach can be traced to the beginning of the decade with work on law-governed systems in the context of software engineering (Minsky, [37]). In this framework, the author studied how a set of rules, governing the behavior of the objects of a system, can facilitate the implementation of protocols. In Minsky's terms a law is "an enforced specification of protocols". The approach is particularly adept at coordinating heterogeneous systems where objects are driven by many programs that can possibly change over time. In such settings, it is highly unrealistic to construct a system that obeys the desired protocol (the so-called constructive implementation of protocols), and an additional layer (the law) is needed. Minsky defines a law as a sequence of primitive actions to be performed in response to the occurrence of a controlled event. Using this definition, he presents various applications of laws whose objective is to enforce communication protocols [37,38].

In the AI community, Moses, Shoham and Tennenholtz [39–42,56] initiated the study of social laws as as an approach to the coordination of multi-agent systems. The approach is broader than in [37] in which a law is a sequence of actions to be taken when an event occurs. In the artificial social systems approach, a social law is a set of constraints on the agents' activity which is common knowledge among the agents. The effect of a social law is twofold. On the one hand, it restricts the freedom of an agent. On the other hand, it reduces the non-determinism which characterizes plans in distributed (and in particular, loosely-coupled) environments. This is exemplified in [56] which studies coordination of multiple robots in a multi-robot grid environment, where different social laws are imposed on the system. The domain consists of an $n \times n$ grid and robots moving on this grid. The authors analyze a number of social laws for this case study. Each guarantees that no collisions will happen and that the robots will be able to design plans to move to any location on the grid. They also analyze the efficiency of these different social laws as a function of the time and the resources spent in the process of moving around, and discuss several computational issues related to the design of the laws. In the model that the authors use, a social law is a set of pairs $\{(a, \mathcal{P})\}$ where a is an action and $\mathcal{P}$ is a predicate on the state of the system. This predicate states the most general condition on the system for which taking a is allowed. Using this model, the authors are able to prove that the design of artificial social laws is, in general, NP-complete. However, an important observation that they bring forth is that since the design of social laws is mostly an off-line activity, being in NP is not a negative result. It suggests that a guess-and-check procedure is a suitable method for designing social laws.

An alternative model for artificial social systems, consisting of a set of *dependent automata*, is presented in [42]. Intuitively, this model defines a multi-agent system as a set of non-deterministic automata. A social law explicitly prohibits certain actions at some of the states of the automata, thereby reducing the non-determinism of the transition functions. In the same study the authors also develop a logical framework to deal with

the semantics of artificial social systems. Modal logic operators allow them to express the notions of ability and constraints and are shown to be useful for reasoning about legally and physically available actions.

Clearly, restricting the actions of the agents without discrimination has little chance to be very helpful to the system. Consider for example a domain consisting of roads on which our agents travel. Those roads cross one another at junctions where total freedom on the side of the agents makes accidents much likely (obviously a very undesirable event for any agent). In order to guarantee accident-free traffic, we could design a social law and oblige all the agents to abide by it. For example, we could set a law that allows our agents to enter the intersection only if the crossing road is free. This law certainly prevents accidents. However, it restricts the agents too much. Although we have guaranteed an accident-free environment, we have also introduced a possibility of a deadlock: when two agents reach the intersection via crossing roads, they might find themselves waiting infinitely for the crossing road to get free before initiating their move. This example illustrates the fact that we can not accept any social law, but must ask that the law be useful, i.e., will enable the agents to achieve their goals. For example, the law could oblige the agents to cross the intersection one at a time, in a round-robbin policy.

The cornerstone of the law-based approach is therefore to benefit the system and guarantee the specification. In other words, by imposing restrictions on the actions the agents can take, the designer guarantees that they will always be able to achieve their goals, no matter how the other agents behave. This property, usefulness, constitutes a clear-cutting criterion for accepting a possible social law. Non-useful social laws are not to be considered. Therefore, although an agent would certainly prefer a law which would restrict the other agents completely while leaving him utterly free (and in this trivial case, there would be no interactions with other agents' plans), such a law would not be useful as the other agents' goals would not be guaranteed. A good social law should therefore restrict the agents' actions in order to minimize the possibility of conflicts, but should also leave these agents enough freedom to achieve their goals. This tradeoff is referred to as the Golden Mean Problem [42], and can be succinctly described as striking the right balance between freedom and constraints.

Subsequent research on social laws has explored both theoretical and applied topics. Tennenholtz [60] studies stable social laws where the notion of stability relies on rationality. Briggs and Cook [6] study the situation where the agents are able to try out different laws (from restrictive to lenient) in order to achieve their goals. They describe an iterative process of deriving plans in social systems where the social law of the system is decided upon dynamically (by communication between the agents or plan synchronization). Although their approach considers several social laws among which agents are choosing, they do not attempt to formally define an order relation on the set of laws or a criterion for choosing a law. Also, the fundamental concepts of usefulness and off-line design are somewhat absent from their study.

Application-oriented research has extensively dealt with the multi-robot grid environment presented in [56] and outlined above. Ben-Yitzhak and Tennenholtz [2,3] derive a method for automatically synthesizing social laws in any grid environment, with or without obstacles, where the purpose of the law is to specify on each segment whether motion

is allowed and if it is, in which direction and at which velocity. The output of the algorithm (the social law) is experimentally tested and compared to alternative approaches.

Moving still a step ahead in the search for automatic synthesis of social laws, Onn and Tennenholtz [44] show how to efficiently derive social laws for motion control of agents in any two-connected graph environment. They define the notion of graph-routing and show that a network whose underlying graph is two-connected has a useful social law if it admits a routing. They prove that any two-connected graph has a routing (which can be efficiently constructed) and therefore, come up with an efficient procedure to design useful social laws in any graph with such a topology.

In this work, we examine two criteria, minimality and simplicity, whose purpose is to help the designer of an artificial social system in choosing among possible laws. The existence of multiple useful social laws with a wide range of different properties motivates this study. Driven by the observation that artificial social systems with fewer restrictions display robustness properties when faced with perturbations, we defined a measure of optimality based on minimal sets of constraints. Our analysis stresses the fact that constraints placed on an artificial social system might not correspond to the constraints actually needed for efficient and flexible behavior. In addition, we have applied the general model to domains of interest, in order to study the role of minimality in a variety of contexts. However, minimality, in its attempt to maximize the set of legal actions, can lead to quite complex social laws. We therefore introduce an alternative criterion, simplicity, which attempts to derive simple social laws based on a complexity measure that we define as well.

Although this paper concentrates on the conceptual and theoretical aspects of selection among social laws, we believe that the insights and concepts presented in this paper can be useful from an implementation perspective as well. Bearing these concepts/criteria in mind when designing new social laws, and proving that these new laws do satisfy these criteria, can lead to better systems and improved agents' behavior. For example, previously hand-crafted social laws lack an appropriate justification for their selection. An analysis of their properties from the perspective of minimality and simplicity may shed light on their properties. The results presented in this paper expose the necessity to foresee future developments and perturbations when designing an artificial social system. Although minimality and simplicity are limited in scope, they appear quite suitable for the off-line design that characterizes artificial social systems, in the sense that they address, at the design stage, issues that the system might face when running. To overcome the intrinsic difficulties with finding a flexible and simple way to the coordination of independent agents, we must continue to investigate the characteristics of systems upon which a law has been imposed, and the conditions under which particular laws have to be preferred or avoided. We expect that further analysis of the influence of different social laws on specific artificial systems may suggest additional promising approaches to classifying artificial social laws. We believe however that the notions of minimal and simple social laws will continue to serve as basic cornerstones in this classification.

References

[1] E. Alonso, Rights and coordination in multi-agent systems, in: Proc. UKMAS-98, Manchester, 1998.
[2] O. Ben-Yitzhak, M. Tennenholtz, On the synthesis of social laws for mobile robots: A study in artificial social systems (Part II), Computers and Artificial Intelligence, to appear.

[3] O. Ben-Yitzhak, M. Tennenholtz, On the synthesis of social laws for mobile robots: A study in artificial social systems (Part I), Computers and Artificial Intelligence 14 (1997).
[4] S. Berthet, Y. Demazeau, O. Boissier, Knowing each other better, in: Y. Demazeau, J.-P. Müller (Eds.), Decentralized AI 2, Elsevier Science, Amsterdam, 1991, pp. 23–42.
[5] A.H. Bond, L. Gasser, Readings in Distributed Artificial Intelligence, Ablex Publishing Corporation, Norwood, NJ, 1988.
[6] W. Briggs, D. Cook, Flexible social laws, in: Proc. IJCAI-95, Montreal, Quebec, 1995, pp. 688–693.
[7] C. Castelfranchi, Social power: A missed point in DAI, MA and HCI, in: Y. Demazeau, J.-P. Müller (Eds.), Decentralized AI, North-Holland/Elsevier, Amsterdam, 1990, pp. 49–62.
[8] C. Castelfranchi, Commitment: From intentions to groups and organizations, in: Proc. 1st International Conference on Multi-Agent Systems, San Francisco, CA, 1996, pp. 41–48.
[9] C. Castelfranchi, E. Werner, Artificial Social Systems, From Reactive to Intentional Agents, 1992.
[10] P.R. Cohen, H.J. Levesque, Teamwork, Nous 25 (4) (1991).
[11] R. Conte, M. Miceli, C. Castelfranchi, Limits and levels of cooperation: Disentangling various types of prosocial interaction, in: Y. Demazeau, J.-P. Müller (Eds.), Decentralized AI 2, North-Holland/Elsevier, Amsterdam, 1991, pp. 147–157.
[12] Y. Demazeau, J.-P. Müller, Decentralized AI, North-Holland/Elsevier, Amsterdam, 1990.
[13] Y. Demazeau, J.-P. Müller, Decentralized AI 2, North-Holland/Elsevier, Amsterdam, 1991.
[14] Y. Demazeau, J.-P. Müller, From reactive to intentional agents, in: Y. Demazeau, J.-P. Müller (Eds.), Decentralized AI 2, North-Holland/Elsevier, Amsterdam, 1991, pp. 3–10.
[15] J. Doyle, M.P. Wellman, Impediments to universal preference-based default theories, in: Proc. 1st Conference on Principles of Knowledge Representation and Reasoning (KR-89), Toronto, Ont., 1989.
[16] E. Durfee, What your computer really needs to know, you learned in kindergarten, in: Proc. AAAI-92, San Jose, CA, 1992, pp. 858–864.
[17] E.H. Durfee, V.R. Lesser, D.D. Corkill, Coherent cooperation among communicating problem solvers, IEEE Trans. Comput. 36 (1987) 1275–1291.
[18] C. Dwork, Y. Moses, Knowledge and common knowledge in a Byzantine environment: Crash failures, Inform. and Comput. 88 (2) (1990) 156–186.
[19] M.J Fischer, N.A. Lynch, M. Paterson, Impossibility of distributed consensus with one faulty processor, in: Proc. 2nd ACM Symposium on Principles of Distributed Computing, 1983.
[20] M.S. Fox, An organizational view of distributed systems, IEEE Trans. Systems Man. Cybernet. 11 (1981) 70–80.
[21] D. Fudenberg, J. Tirole, Game Theory, MIT Press, Cambridge, MA, 1991.
[22] M. Garey, D. Johnson, Computers and Intractability—A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, CA, 1979.
[23] J.N. Gray, A transaction model, in: Lecture Notes in Computer Science, Vol. 85, Springer, Berlin, 1980, pp. 282–298.
[24] D. Grossman, Traffic control of multiple robot vehicles, IEEE J. Robotics and Automation 5 (4) (1988) 491–497.
[25] B.J. Grosz, C.L. Sidner, Plans for discourse, in: P.R. Cohen, J. Morgan, M.E. Pollack (Eds.), Intentions in Communication, MIT Press, Cambridge, MA, 1990, pp. 417–444.
[26] N.R. Jennings, Controlling cooperative problem solving in industrial multi-agent systems using joint intentions, Artificial Intelligence 75 (2) (1995) 195–240.
[27] W.A. Kornfeld, C.E. Hewitt, The scientific community metaphor, IEEE Trans. Systems Man. Cybernet. 11 (1981) 24–33.
[28] S. Kraus, Negotiation and cooperation in multi-agent environments, Artificial Intelligence 94 (1997) 79–97.
[29] S. Kraus, J. Wilkenfeld, The function of time in cooperative negotiations, in: Proc. AAAI-91, Anaheim, CA, 1991, pp. 179–184.
[30] D.M. Kreps, Game Theory and Economic Modelling, Clarendon Lectures in Economics, Clarendon Press, Oxford, 1990.
[31] B. Lampson, H. Sturgis, Crash recovery in a distributed data storage system, Technical Report, Computer Science Laboratory, Xerox, Palo Alto Research Center, Palo Alto, CA, 1976.
[32] A.L. Lansky, Localized event-based reasoning for multiagent domains, Technical Report 423, SRI International, Menlo Park, CA, 1988.

[33] J.C. Latombe, How to move (physically speaking) in a multi-agent world, in: Proc. European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW91), 1991.
[34] T.W. Malone, Modeling coordination in organizations and markets, Management Science 33 (10) (1987) 1317–1332.
[35] J. Marschak, R. Radner, Economic Theory of Teams, Yale University Press, New Haven, CT, 1972.
[36] M. Minsky, The Society of Mind, Simon and Schuster, New York, 1986.
[37] N.H. Minsky, The imposition of protocols over open distributed systems, IEEE Trans. Software Engineering 17 (2) (1991) 183–195.
[38] N.H. Minsky, Law-governed systems, Software Engineering J. (1991) 285–302.
[39] Y. Moses, M. Tennenholtz, Artificial social systems, Part I: Basic principles, Technical Report CS90-12, Weizmann Institute, Rehovot, Israel, 1990.
[40] Y. Moses, M. Tennenholtz, On formal aspects of artificial social systems, Technical Report CS91-01, Weizmann Institute, Rehovot, Israel, 1991.
[41] Y. Moses, M. Tennenholtz, On computational aspects of artificial social systems, in: Proc. 11th Workshop on Distributed Artificial Intelligence, 1992, pp. 267–283.
[42] Y. Moses, M. Tennenholtz, Artificial social systems, Computers and Artificial Intelligence 14 (6) (1995) 533–562.
[43] Y. Moses, M.R. Tuttle, Programming simultaneous actions using common knowledge, Algorithmica 3 (1988) 121–169.
[44] S. Onn, M. Tennenholtz, Determination of social laws for agent mobilization, Artificial Intelligence 95 (1997) 155–167.
[45] M.J. Osborne, A. Rubinstein, A Course in Game Theory, MIT Press, Cambridge, MA, 1994.
[46] A. Pnueli, R. Rosner, Distributed reactive systems are hard to synthesize, in: Proc. 31th IEEE Symposium on Foundations of Computer Science, 1990.
[47] P.G. Ramadge, W.M. Wonham, The control of discrete event systems, Proc. IEEE 77 (1) (1989) 81–98.
[48] J.S. Rosenschein, M.R. Genesereth, Deals among rational agents, in: Proc. IJCAI-85, Los Angeles, CA, 1985, pp. 91–99.
[49] J.S. Rosenschein, G. Zlotkin, Rules of Encounter, MIT Press, Cambridge, MA, 1994.
[50] J.J. Rousseau, Du Contrat Social, Penguin, London, 1762. Translated and introduced by Maurice Cranston (1968). First published in 1762. (Also published by Dent Everyman along with the Discourses.)
[51] A. Rubinstein, Finite automata play the repeated prisoner's dilemma, J. Economic Theory 39 (1986) 83–96.
[52] S. Safra, M. Tennenholtz, On planning while learning, J. Artificial Intelligence Res. 2 (1994) 111–129.
[53] T.W. Sandholm, V.R. Lesser, Equilibrium analysis of the possibilities of unenforced exchange in multiagent systems, in: Proc. IJCAI-95, Montreal, Quebec, 1995, pp. 694–701.
[54] N. Santoro, D. Rotem, On the complexity of distributed elections in synchronous graphs, in: H. Noltemeier (Ed.), International Workshop on Graph-Theoretic Concepts in Computer Science, Trauner Verlag, 1985, pp. 337–346.
[55] Y. Shoham, M. Tennenholtz, On traffic laws for mobile robots, in: Proc. 1st Conference on AI Planning Systems (AIPS-92), 1992.
[56] Y. Shoham, M. Tennenholtz, Social laws for artificial agent societies: Off-line design, Artificial Intelligence 73 (1995) 231–252.
[57] H.A. Simon, The Sciences of the Artificial, MIT Press, Cambridge, MA, 1981.
[58] C.J. Stuart, An implementation of a multi-agent plan synchronizer, in: Proc. IJCAI-85, Los Angeles, CA, 1985, pp. 1031–1033.
[59] G. Tel, Introduction to Distributed Algorithms, Cambridge University Press, Cambridge, 1994.
[60] M. Tennenholtz, On stable social laws and qualitative equilibiria for risk-averse agents, in: Proc. 5th International Conference on Principles of Knowledge Representation and Reasoning (KR-96), Cambridge, MA, 1996.
[61] M. Tennenholtz, On stable social laws and qualitative equilibria, Artificial Intelligence 102 (1998) 1–20.