Offline Norm Evolution

Magnus Hjelmblom[1,2]

[1] Faculty of Engineering and Sustainable Development, University of Gävle, Sweden
mbm@hig.se
[2] Department of Computer and Systems Sciences, Stockholm University, Sweden

Abstract. An approach to the pre-runtime design of normative systems for a class of problem-solving norm-regulated multi-agent systems is suggested. The basic idea is to employ evolutionary mechanisms to evolve efficient normative systems for so-called norm-regulated DALMASes, as part of the design process. The DALMAS architecture uses an algebraic approach to normative systems, in which normative consequences are based on an extended set of one-agent types of normative positions, which is given a semantics in terms of prohibition of certain types of state transitions. To illustrate the approach, a genetic algorithm is used to evolve norms for an example system. Furthermore, some approaches to reducing the algorithm's search space, including to employ a notion of 'operational equivalence' of norms, are discussed. It is demonstrated that an evolutionary algorithm may be a useful tool when designing norms for problem-solving multi-agent systems.
Keywords: Norm-regulated Multi-Agent System, Normative MAS, DALMAS, Norm Evolution, Evolutionary Algorithm

1 Introduction

Agent-based modelling and simulation is an active field of study which, for example, may offer methods for solving complex optimization problems. In this setting, agents are required to cooperate to solve the problem at hand. In complex systems with adjustable agent autonomy, sophisticated planning can often be replaced by norms; see for example [21]. The study of norm-regulated multi-agent systems, often referred to as normative MAS, has also attracted a lot of attention. The NorMAS roadmap [3] is a comprehensive introduction to and overview of the field. The combination of agent-based modelling and simulation and normative MAS is a promising field of study [4].

It is often desirable to replace planning (and replanning), since it may be a complex and time-consuming task, especially in collaborative environments. On the other hand, designing good normative systems is also a challenge. The approach suggested here, whose basic ideas were outlined in [20, pp. 164f], is to use evolutionary mechanisms, employed in a genetic algorithm, to aid the 'off-line' (i.e., pre-runtime) design of normative systems for problem-solving multi-agent systems based on the DALMAS architecture for norm-regulated MAS. The normative framework of a DALMAS is based on an algebraic version of the Kanger-Lindahl theory of normative positions, which is well suited as the logical foundation for normative systems in a MAS context, since the types of normative positions are mutually exclusive and jointly exhaustive in the logical sense.

The paper is structured as follows. Sect. 1.2 briefly introduces the algebraic version of the theory of normative positions, and in Sect. 1.3, previous work on the DALMAS architecture is presented. Sect. 2 introduces an example DALMAS which will be used in Sect. 3 to demonstrate how to employ evolutionary mechanisms in the process of designing norms, by applying an evolutionary algorithm to this example. Sect. 4 concludes and suggests some lines of future work.

1.1 Related Work

The runtime emergence of norms within artificial social systems has attracted the attention of many researchers; see, e.g., [2]. However, evolving normative systems as part of the process of designing norm-regulated MAS is not as well studied, although evolutionary approaches for learning behaviour patterns or strategies for coordination have been successfully used in, e.g., the RoboCup[3] domain; see for example [17,6,18]. In fact, the simple decision policies evolved by Di Pietro et al. for the RoboCup *Keepaway* game can be regarded as simple normative systems consisting of production rules which prescribe certain behaviours in certain situations.

1.2 One-Agent Types of Normative Positions

The Kanger-Lindahl theory of normative positions is based on Kanger's 'deontic action-logic'; see for example [14]. The theory, further developed by Lindahl

[3] `http://www.robocup.org`

Table 1. One-agent types of normative positions

$\mathbf{T}_1(x,F)$: May $\mathrm{Do}(x,F)$ & May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_2(x,F)$: May $\mathrm{Do}(x,F)$ & May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & $\neg$May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_3(x,F)$: May $\mathrm{Do}(x,F)$ & $\neg$May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_4(x,F)$: $\neg$May $\mathrm{Do}(x,F)$ & May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_5(x,F)$: May $\mathrm{Do}(x,F)$ & $\neg$May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & $\neg$May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_6(x,F)$: $\neg$May $\mathrm{Do}(x,F)$ & May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & $\neg$May $\mathrm{Do}(x,\neg F)$
$\mathbf{T}_7(x,F)$: $\neg$May $\mathrm{Do}(x,F)$ & $\neg$May$[\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)]$ & May $\mathrm{Do}(x,\neg F)$

[15], contains three systems of types of normative positions, based on the logic of the action operator Do and the deontic operator Shall. The simplest of these systems is a system of seven 'one-agent types' of normative positions. $\mathrm{Do}(x,F)$ is commonly read as 'x sees to it that F' or 'x brings it about that F', where F is a proposition regarding some state of affairs. The logical properties assumed for Do is that it is the smallest system containing propositional logic, closed under logical equivalence and containing the axiom schema $\mathrm{Do}(x,F) \rightarrow F$, which tries to capture the notion of *successful* action; if x 'sees to it' or 'brings about' that F, then F is indeed the case.

Each of the three statements

(i) $\mathrm{Do}(x,F)$,
(ii) $\mathrm{Do}(x,\neg F)$, and
(iii) $\neg\mathrm{Do}(x,F)$ & $\neg\mathrm{Do}(x,\neg F)$,

implies the negation of each of the others, and the disjunction of all three is a tautology. Each of (i) - (iii) can be prefixed with either May or $\neg$May, where May F is defined as $\neg\mathrm{Shall}\neg F$, and basic conjunctions containing one statement from each such pair can be formed. By iterated construction of basic conjunctions, a set of eight conjunctions (of which one is self-contradictory) is obtained. The consistent 'maxi-conjunctions' are listed in Table 1.

In a series of papers, comprehensively summarized in [16], Lindahl and Odelstad have combined the theory of normative positions with an algebraic approach to normative systems. Their idea is to use the one-agent types of normative positions as operators on descriptive *conditions* to get deontic conditions. A ν-ary condition d can be true or false of ν agents $x_1,...,x_\nu$. Thus, $d(x_1,...,x_\nu)$ is a state of affairs which may be true or false. (To facilitate the presentation, X_ν will often be used as an abbreviation for the argument sequence $x_1,...,x_\nu$.) In the special case when the sequence of agents is empty, i.e. $\nu = 0$, d represents a proposition which may be true or false. Note that negations d', conjunctions $(c \wedge d)$, and disjunctions $(c \vee d)$ can be formed in the following way:

$d'(X_\nu)$ iff $\neg d(X_\nu)$,
$(c \wedge d)(X_\nu)$ iff $[c(X_p)$ and $d(X_q)]$, and
$(c \vee d)(X_\nu)$ iff $[c(X_p)$ or $d(X_q)]$,

Table 2. 'Reduced extended' types of one-agent normative positions

$\mathbf{P}_1(x,F)$: May $\mathrm{Do}(x,F)$ & May $\mathbf{\Lambda}(x,F)$ & May $\mathbf{\Omega}(x,F)$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{P}_{2\Lambda}(x,F)$: May $\mathrm{Do}(x,F)$ & May $\mathbf{\Lambda}(x,F)$ & $\neg$May $\mathbf{\Omega}(x,F)$ & $\neg$May $\mathrm{Do}(x,\neg F)$
$\mathbf{P}_{2\Omega}(x,F)$: May $\mathrm{Do}(x,F)$ & $\neg$May $\mathbf{\Lambda}(x,F)$ & May $\mathbf{\Omega}(x,F)$ & $\neg$May $\mathrm{Do}(x,\neg F)$
$\mathbf{P}_{4\Lambda}(x,F)$: $\neg$May $\mathrm{Do}(x,F)$ & May $\mathbf{\Lambda}(x,F)$ & $\neg$May $\mathbf{\Omega}(x,F)$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{P}_{4\Omega}(x,F)$: $\neg$May $\mathrm{Do}(x,F)$ & $\neg$May $\mathbf{\Lambda}(x,F)$ & May $\mathbf{\Omega}(x,F)$ & May $\mathrm{Do}(x,\neg F)$
$\mathbf{P}_5(x,F)$: Shall $\mathrm{Do}(x,F)$
$\mathbf{P}_{6\Lambda}(x,F)$: Shall $\mathbf{\Lambda}(x,F)$
$\mathbf{P}_{6\Omega}(x,F)$: Shall $\mathbf{\Omega}(x,F)$
$\mathbf{P}_7(x,F)$: Shall $\mathrm{Do}(x,\neg F)$

where $\nu = \max(p,q)$.[4] Therefore, it is possible to construct Boolean algebras of conditions. A Boolean algebra together with an implicative relation R fulfilling certain conditions, forms a so-called *Boolean quasiordering* (*Bqo*). As an application of their Theory of Joining-Systems (TJS), Lindahl and Odelstad define the notion of a *normative position condition-implication structure*, abbreviated *np-cis*, which is based on *Bqo*'s on descriptive and deontic conditions, so-called *cis-Bqo*'s. For details on Boolean quasiorderings, condition implication structures and *np-cis*'es, see for example [16] or [20].

1.3 Previous Work

DALMAS [20] is an abstract architecture for a class of norm-regulated multi-agent systems. A deterministic DALMAS is a simple multi-agent system in which the actions of an agent are connected to transitions between system states. In a deterministic DALMAS the agents take turns to act; only one agent at a time may perform an action. By allowing 'do nothing' actions and accelerating the turn-taking, systems with close to asynchronous behaviour can be obtained. A special kind of DALMAS is the *norm-regulated simple deterministic* DALMAS, which employs what is often referred to as 'negative permission', by letting its *deontic structure* (i.e., the set of permissible acts) consist of all acts that are not explicitly prohibited by a normative system $\mathcal{N}$. The DALMAS's *preference structure* consists of the most preferable (according to the agent's utility function) of the acts in the deontic structure. In short, a DALMAS agent's behaviour is regulated by the combination of a normative system and a utility function; this 'agent oeconomicus norma'[5] chooses the most desirable act, according to the utility function, within the 'room for manoeuvre' determined by the norms. The DALMAS's normative framework is based on an algebraic version of the Kanger-Lindahl theory of normative positions, in which normative consequences

[4] The free variables in $c(x_1,...,x_p)$ must be the same, and in the same order, as the free variables in $d(x_1,...,x_q)$, but it is not necessary that c and d have the same arity. Cf. [20, p. 146].
[5] Cf. [19, Sect. 1.8.3].

are formulated by applying normative operators to descriptive conditions. (See Sect. 1.2.) From these general normative conditions follow normative sentences regarding specific states of affairs, which in turn result in permission or prohibition of individual actions in specific situations. (See for example [20,19] for an introduction.) Hence, the norms in the DALMAS architecture play a different role, and is represented in a fundamentally different way, than, e.g., the decision rules in the *RoboCup* setting (see Sect. 1.1).

Since the agents in a deterministic DALMAS take turns to act, each individual step in a run of a DALMAS may be characterized by an ordered 5-tuple $S = \langle x, s, A, \Omega, S\rangle$ whose components are a set of states S, a state s, an agent-set $\Omega = \{x_1, ..., x_n\}$, the acting ('moving') agent x, and an action-set $A = \{a_1, ..., a_m\}$.[6] In this setting, a may be regarded as a function such that $a(x, s) = s^+$ means that s^+ is the resulting state when x performs act a in state s. In the following, the abbreviation s^+ will be used for $a(x, s)$ when there is no need for an explicit reference to the action a and the acting agent x. As already mentioned, there is no simultaneous action by other agents (including the 'environment', which may be regarded as a special kind of agent). Furthermore, we assume that a ν-ary condition d is true or false on ν agents $x_1, ..., x_\nu \in \Omega$ in s; with the abbreviation X_ν for the agent sequence, this will be written $d(X_\nu; s)$.

Let the situation $\langle x, s\rangle$ be characterized by the moving agent x and the state s in a norm-regulated simple deterministic DALMAS. It is possible in the DALMAS architecture to distinguish between the moving agent and the agent to which normative condition applies[7], but to facilitate the presentation it is assumed in the sequel that norms always apply to the moving agent x in a situation $\langle x, s\rangle$. A norm in $\mathcal{N}$ is represented by an ordered pair $\langle c, Nd\rangle$, where the (descriptive) condition c on a situation $\langle x, s\rangle$ is the *ground* of the norm and the (normative) condition Nd on $\langle x, s\rangle$ is its *consequence*; see, e.g., [20]. Nd is formed by applying a 'norm-creating' operator N to the descriptive condition d.

Table 3. Basic transition types

I.	$d(X_\nu; s)\,\&\,d(X_\nu; s^+)$
II.	$\neg d(X_\nu; s)\,\&\,d(X_\nu; s^+)$
III.	$d(X_\nu; s)\,\&\,\neg d(X_\nu; s^+)$
IV.	$\neg d(X_\nu; s)\,\&\,\neg d(X_\nu; s^+)$

In the following, the normative framework of the DALMASes employed is based on the notion of an *np9-cis* [12, Sect. 2.2.1], a structure similar to the *np-cis* defined by Lindahl and Odelstad, but based on the 'reduced extended' set of types of normative positions shown in Table 2. It is argued in [12] that a semantics for the normative framework of a DALMAS can be formed by defining a set of 'transition type operators' C^a_k, $k \in \{1, 2\Lambda, 2\Omega, 4\Lambda, 4\Omega, 5, 6\Lambda, 6\Omega, 7\}$, based

[6] In [9], such a tuple is called a *transition system situation.*

[7] Cf. the remark in [10, p. 84].

on Table 4, and a set of corresponding 'transition type prohibition operators' P_k, such that $P_k d(X_\nu; x, s)$ is intended to mean that if $C_k^a d(X_\nu; x, s)$ holds, then a is prohibited for x in $\langle x, s\rangle$. In effect, $P_k d(X_\nu; x, s)$ implies a prohibition of zero, one or two of the four 'basic transition types' (see Table 3, where s^+ refers to the resulting state when the acting agent performs its act) with regard to the state of affairs $d(X_\nu)$. For example, $\langle c, P_k d\rangle$, where c and d can have different arity, represents the sentence

$$\forall x_1, x_2, ..., x_\nu \in \Omega : c(x_1, x_2, ..., x_p; x, s) \rightarrow P_k d(x_1, x_2, ..., x_q; x, s)$$

where Ω is the set of agents, x is the acting agent (to which the norm applies) in the situation $\langle x, s\rangle$, and $\nu = \max(p, q)$. If the condition specified by the ground of a norm for some agents in some situation, then the (normative) consequence of the norm is in effect in that situation. If the normative system $\mathcal{N}$ contains a norm whose ground holds in the situation $\langle x, s\rangle$ and whose consequence prohibits the type of transition represented by x performing action a, then a is prohibited for x in $\langle x, s\rangle$:

$$\begin{gathered} \mathit{Prohibited}_{x,s}(a) \text{ according to } \mathcal{N} \\ \text{if there is a } p\text{-ary condition } c \\ \text{and a } q\text{-ary condition } d \\ \text{and a } k \in \{1, 2\Lambda, 2\Omega, 4\Lambda, 4\Omega, 5, 6\Lambda, 6\Omega, 7\}, \\ \text{such that } \langle c, P_k d\rangle \text{ is a norm in } \mathcal{N}, \\ \text{and there are } x_1, ..., x_\nu \text{ such that} \\ c(x_1, ..., x_p; x, s) \ \&\ C_k^a d(x_1, ..., x_q; x, s), \\ \text{where } \nu = \max(p, q). \end{gathered}$$

Hence, if $c(x_1, ..., x_p; x, s)$ for some sequence of agents $x_1, ..., x_\nu$, then the normative condition $P_k d(x_1, ..., x_q; x, s)$ is 'in effect'. Thus, if $C_k^a d(x_1, ..., x_q; x, s)$ holds, then a is prohibited for x in s. (Cf. the examples in Sect. 3.1.) Table 4 contains the nine norm-building operators P_k, together with the corresponding C_k^a operators and the result of applying them to $d(x_1, ..., x_q; s)$; cf. Table VI in [11].

A general-level Java/Prolog implementation of the DALMAS architecture has been developed, to facilitate the implementation of specific systems. The Colour & Form system, the Waste-collector system and the Forest Cleaner system are three specific systems that have been implemented using this framework. The reader is referred to [20,7,13,8] for a description of these systems and their instrumentalizations.

The approach to normative systems employed in this framework is ideally suited for evolution of normative systems, since the nine 'reduced extended' types of normative positions are mutually exclusive and jointly exhaustive in the logical sense. Therefore each conceivable normative system, consisting of conditional norms based on descriptive conditions selected from a set of potential grounds and normative conditions selected from a set of potential consequences, could become a candidate for evaluation in the execution of an evolutionary algorithm. This idea will be further explored in the following sections.

Table 4. Transition type prohibition operators and transition type conditions

Operators		$Prohibited_a(x,s)$ if
P_1	-	-
$P_{2\Lambda}$	$C^a_{2\Lambda}$	$d(X_\nu;s)\,\&\,\neg d(X_\nu;a(x,s))$
$P_{2\Omega}$	$C^a_{2\Omega}$	$\neg d(X_\nu;s)\,\&\,\neg d(X_\nu;a(x,s))$
$P_{4\Lambda}$	$C^a_{4\Lambda}$	$\neg d(X_\nu;s)\,\&\,d(X_\nu;a(x,s))$
$P_{4\Omega}$	$C^a_{4\Omega}$	$d(X_\nu;s)\,\&\,d(X_\nu;a(x,s))$
P_5	C^a_5	$\neg d(X_\nu;a(x,s))$
$P_{6\Lambda}$	$C^a_{6\Lambda}$	$\neg(d(X_\nu;s)\leftrightarrow d(X_\nu;a(x,s)))$
$P_{6\Omega}$	$C^a_{6\Omega}$	$d(X_\nu;s)\leftrightarrow d(X_\nu;a(x,s))$
P_7	C^a_7	$d(X_\nu;a(x,s))$

Table 5. Possible changes of Lap_n

State of affairs	Possible state of affairs in next state
$Lap_0(x_1,x_2)$	$Lap_0(x_1,x_2), Lap_1(x_1,x_2), Lap_2(x_1,x_2), Lap_3(x_1,x_2)$
$Lap_1(x_1,x_2)$	$Lap_0(x_1,x_2), Lap_1(x_1,x_2), Lap_2(x_1,x_2)$
$Lap_2(x_1,x_2)$	$Lap_0(x_1,x_2), Lap_1(x_1,x_2), Lap_2(x_1,x_2), Lap_3(x_1,x_2), Lap_4(x_1,x_2)$
$Lap_3(x_1,x_2)$	$Lap_0(x_1,x_2), Lap_2(x_1,x_2), Lap_3(x_1,x_2), Lap_4(x_1,x_2), Lap_6(x_1,x_2)$
$Lap_4(x_1,x_2)$	$Lap_2(x_1,x_2), Lap_4(x_1,x_2), Lap_6(x_1,x_2)$
$Lap_6(x_1,x_2)$	$Lap_3(x_1,x_2), Lap_4(x_1,x_2), Lap_6(x_1,x_2), Lap_9(x_1,x_2)$
$(=' \wedge Lap_9)(x_1,x_2)$	$Lap_6(x_1,x_2), Lap_9(x_1,x_2)$

2 Example: Explorer DALMAS

Let us consider a class of systems of agents operating in an environment consisting of a grid of squares ordered in rows and columns, in which each square is assigned a pair of integer coordinates. Let us assume that the joint goal of the agents is to explore as much as possible of the grid using a fixed number of moves. An agent can *stay* in the current square, i.e., do nothing, or *move* one square in one of four directions (east, north, west, south) as long as it stays within the boundaries of the grid. In other words, in a given situation, an action is feasible if and only if it does not move the agent off limits. It should of course be noted that these simple systems (in the following referred to as Explorer DALMASes) in themselves are of limited interest, but the idea here is to illustrate how evolutionary mechanisms could be used in the process of designing normative systems for problem-solving MAS.

To simulate a situation with limited possibilities for communication between agents and only local knowledge of the environment, we further assume that an agent only knows the status (visited or unvisited) of the immediately surrounding squares, and the location of other agents within two squares. An agent's preference is represented by a very simple utility function such that moving to an unvisited square is preferred over moving to a visited square, and stay is the

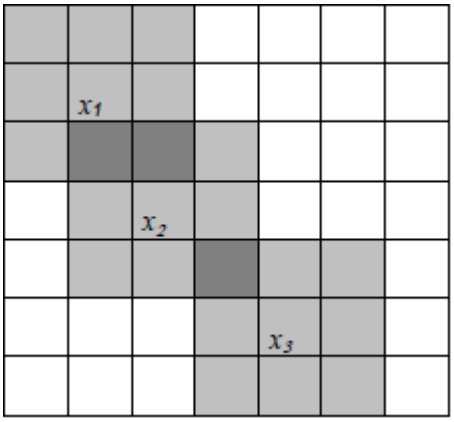

Fig. 1. Overlap of the protected spheres for three agents

least preferred action. In the case of a tie between equally preferred actions, one of them is randomly selected. This means that all agents have the same utility function.

To make the situation more concrete, let us assume that the size of the grid is $c \times r$ squares and place three agents at square $(1, 1)$, the leftmost lowest square. Note that this system can be considered as an instance of the Waste-collector system [20,13], in which visited (resp., unvisited) squares are represented by 0 (resp., 1) units of 'waste'. The higher number of 'waste' carried by an agent, the higher number of unvisited squares have been entered by that agent. It would not be a very difficult task to design a plan where the agents take turns to act in such a way that all remaining $cr - 1$ squares are visited in $cr - 1$ moves. But if the environment gets changed, e.g., is resized or reshaped, the plan must be recalculated. What if we let norms replace plans in this class of environments? Let us investigate the interplay between the agents' utility functions, representing their 'desires', and a normative system which determines their 'room for manoeuvre'. One idea is to base norms on the spatial relationship between the agents, potentially restricting how the agents may move in the proximity of other agents. We define the condition Lap_n, $n \in \{0, 1, 2, 3, 4, 6, 9\}$, with the intended meaning that $Lap_n(x_i, x_j; s)$ holds if and only if the *protected spheres* of agents x_i and x_j overlap with n squares in a state s. The protected sphere consists of the agent's square plus the eight surrounding squares; see Fig. 1, which illustrates a state in which $Lap_2(x_1, x_2)$, $Lap_1(x_2, x_3)$, and $Lap_0(x_1, x_3)$ holds. Table 5 shows how the overlap can change from one state to another, given the five available actions. Note that $Lap_n(x_i, x_j)$ implies $x_i =' x_j$ for $n < 9$, and $x_i = x_j$ implies $Lap_9(x_i, x_j)$. Furthermore, $Lap_n(x_i, x_j)$ implies $\neg Lap_m(x_i, x_j)$ for $n \neq m$. In other words, $Lap_n \, R ='$ for $n < 9$, $= R \, Lap_9$, and $Lap_n \, R \, Lap'_m$ for $n \neq m$, where d' is the negation of the condition d and R is the implicative relation on conditions which defines the *cis-Bqo*'s of grounds and consequences of the normative system.

Now let the 'elementary' conditions Lap_0, Lap_1, Lap_2, Lap_3, Lap_4, Lap_6, together with the 'non-elementary' condition $(=' \wedge Lap_9)$, form a set of potential descriptive grounds for conditional norms. The set of potential normative consequences corresponding to each ground is constructed by applying the norm-building operators P_1, P_{2A}, ..., P_7 (see Sect. 1.3) to the conditions listed in the corresponding rows in Table 5. Thus, the potential consequences for, e.g., Lap_1 are P_1Lap_0,..., P_7Lap_0, P_1Lap_1,..., P_7Lap_1, and P_1Lap_2,..., P_7Lap_2. Note that it would be meaningless to, e.g., let P_iLap_4 be a potential consequence for Lap_0, since none of the available acts can change the state of the system in such a way that $Lap_0(x_i, x_j)$ holds in one state and $Lap_4(x_i, x_j)$ holds in the next state.

With these building blocks available, normative systems for Explorer DALMASes can be constructed. Let us employ the following scheme: For each condition c in the leftmost column of Table 5, one norm $\langle M_1c, P_id\rangle$ is added to the normative system for each condition d in the rightmost column.[8] E.g., for Lap_0 we add four norms: $\langle M_1Lap_0, P_{k_0}Lap_0\rangle$, $\langle M_1Lap_0, P_{k_1}Lap_1\rangle$, $\langle M_1Lap_0, P_{k_2}Lap_2\rangle$, and $\langle M_1Lap_0, P_{k_3}Lap_3\rangle$. Note that, as regards the ground $(=' \wedge Lap_9)$, one of $\langle M_1(=' \wedge Lap_9), P_{k_0}Lap_9\rangle$ and $\langle M_1(=' \wedge Lap_9), P_{k_1}Lap_6\rangle$ is redundant, and can therefore be removed.[9] This gives a total of 24 norms. Note, however, that not all normative systems formed in this way are *coherent*. To begin with, some sets of rules may be contradictory, according to the intended meaning of the P_i operators, but the problem of coherence (sometimes referred to as 'absence of conflicts') cannot simply be reduced to logical consistency; see for example [1]. We will return to this issue in Sect. 3.1.

We would now like to find the best normative system, i.e., the normative system that, together with the simple utility function described earlier, on average makes the Explorer system most efficient. The following measure of efficiency will be employed: the normative system is applied to three different Explorer DALMASes, operating on grids of (almost) equal sizes but different shapes: 6×8 squares, 7×7 squares, and 10×5 squares, respectively. On each grid, three agents are initially placed on square $(1, 1)$. A k-event run of each of these three systems will be performed, where k is the number of unvisited squares from the beginning, i.e., $k = cr - 1$. For each run, the ratio between the total number of visited squares and the total number of unvisited squares in the beginning is calculated. If the normative system is not coherent, in the sense that, at some point during the run, all actions (including stay) become prohibited for the acting agent, then the evaluation score is set to 0. The score of the normative system under evaluation is then the average of the three ratios obtained. We have now obtained

[8] The 'move operator' M_κ, where κ is less than or equal to the arity of the condition to which it is applied, identifies the agent to which the normative condition applies with the moving agent x in the situation $\langle x, s\rangle$, as well as with the κth agent in the argument sequence X_ν. For example, $M_1Lap_0(x_1, x_2, x_3; x, s)$ holds if and only if $Lap_0(x_1, x_2; s)$, and $x_1 = x_3$, and $x_3 = x$. See, e.g., [9] for an explanation.

[9] This is due to the fact that $Lap_6\, R\, Lap_9'$ and $Lap_9\, R\, Lap_6'$. Thus, if a certain type of normative position holds regarding Lap_9, then this completely determines the type of normative position regarding Lap_6, or vice versa. For example, when Lap_9 holds, if P_7Lap_6, then it must follow that P_5Lap_9.

an optimization problem which may be solved with the help of an evolutionary algorithm.

3 Evolution of Explorer Norms

Evolutionary algorithms (EA), being a subfield of evolutionary computation, use the principles of biological evolution (such as reproduction, mutation, recombination, and selection) to solve problems on computers. For a comprehensive introduction to this field the reader is referred to, e.g., [22]. In the Explorer DALMAS setting, there is some randomness in the agents' choices of actions, and in such 'noisy' domains, evolutionary algorithms are known to work well [5]. We thus implement a basic genetic algorithm (one of the most common forms of EAs) for Explorer DALMAS norms:

1. **Genesis**
   Create an initial population of n candidate normative systems, half of which are entirely randomly generated and half of which consist of P_1-consequences (the most permissive consequences) only. Each candidate is represented by a character string consisting of 24 characters, one for each norm, from {'1', ...,'9'}, where '1' represents P_1, '2' represents $P_{2\Lambda}$, '3' represents $P_{2\Omega}$, '4' represents $P_{4\Lambda}$, etc.
2. **Evaluation**
   Evaluate each member of the population, by translating the character string to a normative system according to the scheme presented in Sect. 2, running three different systems regulated by this normative system and using as fitness function the average of the evaluation scores of the three runs. For example, a 'chromosome' starting with "371529..." is translated to the following normative system:
   $\{\langle M_1(=' \wedge Lap_9), P_{2\Omega} Lap_9\rangle, \langle M_1 Lap_6, P_{6\Lambda} Lap_9\rangle, \langle M_1 Lap_6, P_1 Lap_6\rangle, \langle M_1 Lap_6, P_{4\Omega} Lap_4\rangle, \langle M_1 Lap_6, P_{2\Lambda} Lap_3\rangle, \langle M_1 Lap_4, P_7 Lap_6\rangle, \ldots\}$
3. **Survival of the Fittest**
   Select a number of members of the evaluated population, favouring those with higher fitness scores, to be the parents of the next generation.
4. **Evolution**
   Generate a new population of offspring by randomly altering and combining elements of the parent candidates. The evolution is performed by the two basic evolutionary operators *cross-over* and *mutation*.
5. **Iteration**
   Repeat steps 2-4 until the termination condition (see Table 6) is met.

The evolutionary algorithm was implemented using the Java-based Watchmaker framework for evolutionary computation[10] together with a slightly adapted Java/Prolog implementation of the Waste-collector system [7,13].[11] The latter

[10] `http://watchmaker.uncommons.org/`

[11] The source code is available for download via `http://drpa.se/norms/nrtssit`, together with a log of a run of the algorithm.

Fig. 2. Evolution progress. Upper curve shows fitness of best individual, lower curve shows mean fitness. Algorithm parameter values are shown in Table 6. A log of the run is available for download via `http://drpa.se/norms/nrtssit/`.

was used in step 2 to perform the k-event runs of Explorer systems to be evaluated.

3.1 Result

The algorithm was run with the parameter values shown in Table 6; the execution time on an ordinary laptop was 5-6 hours. The graph in Fig. 2 shows the fitness values (evaluation scores) of the best normative system, as well as the average fitness values, in each generation. We can see that, initially, the best fitness (which is obtained by a normative system with P_1-consequences only, i.e., a normative system which allows everything) is around 0.78. Up to around generation 25, we can see a slow but quite steady improvement in the best fitness values, although the impact of the slight randomness in the agents' choices of actions is clear. The highest scores, just above 0.86, which roughly corresponds to three more visited squares per run, are obtained in generations 41 and 78. After 25 generations there seems to be no significant improvement.

According to the log, the best normative system in generation 41 (with P_1-norms omitted for brevity) is translated to

$$\langle M_1(=' \wedge Lap_9), P_{2\Omega} Lap_9\rangle, \langle M_1 Lap_6, P_{6\Lambda} Lap_9\rangle, \langle M_1 Lap_4, P_{6\Omega} Lap_4\rangle,$$
$$\langle M_1 Lap_3, P_{2\Lambda} Lap_6\rangle, \langle M_1 Lap_2, P_{6\Lambda} Lap_4\rangle, \langle M_1 Lap_2, P_{4\Omega} Lap_3\rangle,$$
$$\langle M_1 Lap_1, P_{4\Omega} Lap_2\rangle, \langle M_1 Lap_0, P_{4\Lambda} Lap_1\rangle.$$

Table 6. Choice of Parameter Values

Parameter	**Value**
Population size	100 individuals
Termination condition	100 generations evolved
Level of elitism	25%
Crossover probability	0.7
Crossover points	6
Mutation probability	0.05
Selection strategy	Roulette wheel selection

A closer look at the log reveals that, of the best candidates with a fitness over 0.85,

(a) all but one (13 out of 14) contain either $\langle M_1 Lap_6, P_{6\Lambda} Lap_9 \rangle$ or $\langle M_1 Lap_6, P_{4\Lambda} Lap_9 \rangle$, and
(b) all but three contain $\langle M_1 Lap_2, P_{6\Lambda} Lap_4 \rangle$ or $\langle M_1 Lap_2, P_{4\Lambda} Lap_4 \rangle$.

Let us first consider (a). The intended meaning of $\langle M_1 Lap_6, P_{6\Lambda} Lap_9 \rangle$ is that if $Lap_6(x_1, x_2; s)$ for some agents x_1, x_2, and x_3 such that $x_1 = x_3$ and x_3 is the moving agent x, then the normative condition $P_{6\Lambda} Lap_9(x_1, x_2, x; x, s)$ holds, and thus, according to the definition of $C^a_{6\Lambda}$ (see row 7 in Table 4) action a is prohibited for x if

$$\neg Lap_9(x, x_2; s) \,\&\, Lap_9(x, x_2; a(x, s))$$

or

$$Lap_9(x, x_2; s) \,\&\, \neg Lap_9(x, x_2; a(x, s)).$$

Now, since $Lap_6 \, R \, Lap'_9$ (i.e., Lap_6 implies Lap'_9), the second disjunct never becomes true when $Lap_6(x, x_2; s)$; hence a is prohibited for x if

$$\neg Lap_9(x, x_2; s) \,\&\, Lap_9(x, x_2; a(x, s)).$$

Thus, if $Lap_9(x, x_2; a(x, s))$ then a is prohibited for x, since $Lap_6(x, x_2; s)$ implies $\neg Lap_9(x, x_2; s)$. Similarly, the meaning of $\langle M_1 Lap_6, P_{4\Lambda} Lap_9 \rangle$ is that if $Lap_6(x_1, x_2; s)$ for some agents x_1, x_2, and x_3, such that $x_1 = x_3$ and $x_3 = x$, then $P_{4\Lambda} Lap_9(x_1, x_2, x; x, s)$ holds, and thus (see Table 4, row 4) a is prohibited for x if

$$\neg Lap_9(x, x_2; s) \,\&\, Lap_9(x, x_2; a(x, s));$$

i.e., when $Lap_6(x, x_2; s)$, it follows that $\neg Lap_9(x, x_2; s)$, and thus a is prohibited for x if $Lap_9(x, x_2; a(x, s))$. Hence, the norms $\langle M_1 Lap_6, P_{6\Lambda} Lap_9 \rangle$ and $\langle M_1 Lap_6, P_{4\Lambda} Lap_9 \rangle$ are 'operationally equivalent' in the Explorer DALMAS setting, in the sense that they prohibit the same actions in the same situation.

Furthermore, both are operationally equivalent to $\langle M_1 Lap_6, P_7 Lap_9 \rangle$ with the intended interpretation that if Lap_6 then the moving agent shall see to it that not Lap_9. A similar case can be made for (b); the norms $\langle M_1 Lap_2, P_{6\Lambda} Lap_4 \rangle$ and $\langle M_1 Lap_2, P_{4\Lambda} Lap_4 \rangle$ (and $\langle M_1 Lap_2, P_7 Lap_4 \rangle$) are operationally equivalent and thus interchangeable in this setting. The notion of operational equivalence of norms is further discussed in Sect. 3.2.

It is showed by (a) and (b) that, in some settings, the set of consequences may contain redundancy. This is an effect of the fact that, in this particular setting, the set of grounds and the set of consequences are constructed from the same set of conditions. Whether this is a problem or not is probably dependent on the particular setting. We may also note that, for example, the meaning of $\langle M_1 Lap_0, P_{4\Omega} Lap_2 \rangle$ would be that if $Lap_0(x_1, x_2; s)$ for some agents x_1, x_2, and x_3, such that $x_1 = x_3$ and $x_3 = x$, then $P_{4\Omega} Lap_2(x, x_2, x; x, s)$ holds, and thus (see Table 4, row 5) a is prohibited for x if

$$Lap_2(x, x_2; s) \,\&\, Lap_2(x, x_2; a(x, s)).$$

Now, since $Lap_0 \, R \, Lap_2'$, $Lap_2(x, x_2; s) \,\&\, Lap_2(x, x_2; a(x, s))$ can never become true when $Lap_0(x, x_2; s)$. Hence, $\langle M_1 Lap_0, P_{4\Omega} Lap_2 \rangle$ will never prohibit any actions, and is thus operationally equivalent to, $\langle M_1 Lap_0, P_1 Lap_2 \rangle$ in this setting. This illustrates another kind of redundancy. Another consequence of employing negative permission is that normative systems may evolve which are incoherent (see Sect. 1.3) according to the underlying logic of the P_k operators, but still meaningful in an 'operational' sense. A discussion of these matters is beyond the scope of this paper, but a more precise representation of genes and a more careful design of the genetic operators could avoid or at least reduce logical incoherence and redundancy in the setting at hand. This could, potentially, significantly reduce the search space for the evolutionary algorithm. For this purpose, the mechanisms for norm addition and subtraction described in [16, Sect. 4.3] might be very useful, as well as a more thorough analysis of the relationships between potential grounds and consequences, in order to to exploit the possibility of operational equivalence of norms (Sect. 3.2).

Based on the above analysis, the following set of Explorer norms (again, P_1-norms are omitted) is suggested: $\{\langle M_1 Lap_6, P_7 Lap_9 \rangle, \langle M_1 Lap_2, P_7 Lap_4 \rangle\}$. The intended interpretation is

(1) $\forall x, y : Lap_6(x, y; s) \rightarrow P_7 Lap_9(x, y, x; x, s))$; and
(2) $\forall x, y : Lap_2(x, y; s) \rightarrow P_7 Lap_4(x, y, x; x, s))$.

Using the deontic operator Shall and the action operator Do, these norms are expressed as follows: (1) For all x, y: if $Lap_6(x, y)$, and x is the moving agent, then $\mathrm{Shall}\,\mathrm{Do}(x, \neg Lap_9(x, y))$; and (2) For all x, y: if $Lap_2(x, y)$, and x is the moving agent, then $\mathrm{Shall}\,\mathrm{Do}(x, \neg Lap_4(x, y))$; cf. [20,11]. This represents the following simple set of 'rules of thumb': (1) If you stand in the square next to another agent's square, you shall act so that you do not end up in the same location as the other agent, and (2) if your protected sphere overlaps another

agent's protected sphere with two squares, you shall act so that the overlap does not increase to four.

Test runs indicate that the average improvement with this very simple normative system compared with a system with no restrictions is two to three additional squares visited. As the Explorer DALMAS example was chosen primarily for demonstration purposes, we shall be content with the simple analysis performed here. In more complex scenarios, other more powerful (e.g., statistical) methods could be useful.

3.2 Discussion

Validation of the suggested approach to the design of normative systems for problem-solving MAS is, of course, a non-trivial problem. One aspect of this problem is the difficulty of applying this approach, but most important is probably to focus on the quality of the results it produces, i.e., to validate the systems obtained by applying the approach. The performance of norm-regulated MAS designed in this way could, for example, be compared with the performance of systems (norm-regulated systems as well as, e.g., planning systems) designed by hand. Such comparisons require domain-specific performance measures, which makes a general-level (i.e., domain independent) validation very difficult, if not impossible. Even within a specific domain, validation is non-trivial and sensitivity analyses are required. A good starting-point is to consider every tool in the evolutionary toolbox, together with a thorough analysis of the domain at hand, to increase the chance of evolving the optimal normative system. First, the parameters controlling the evolutionary algorithm may be varied: the population size, the number of evolved generations, the level of elitism (i.e., the portion of the best candidates which are allowed to survive into the next generation), the probability of crossover, the number of crossover points, and the selection strategy (e.g., tournament selection instead of roulette wheel). Other ideas include using other representations of chromosomes, such as tree-based representations to allow for normative systems with a variable number of norms, or (as has already been mentioned) more carefully designed evolutionary operators that exclude redundant and/or incoherent candidates from evaluation. More advanced schemes, such as island evolution (where several populations are evolved in parallel, with a small probability of 'migration' between such 'islands') or cooling (where the crossover and mutation probabilities gradually decrease), could also be tried. One example of a more careful design of an evolutionary operator is to restrict the mutation operator by the notion of 'deontic paths' [15, pp. 110ff] between types of normative positions. In short, the deontic path follows the edges in the Hasse diagram of the relation 'less free than' on types of normative positions; cf. [15, p. 105] and [12, Fig. 1]. The restriction could be that a gene which represents a type operator P_i applied to a descriptive condition d, may only be changed by mutation to represent a new operator P_j in such a way that P_j lies immediately above or below P_i on the deontic path between them. This could bring more stability into the evolution process, since the effects of mutations would be less dramatic.

Furthermore, the parameters for the particular setting may also be varied. For example, one might want to consider grounds and consequences based on other conditions. In the Explorer DALMAS domain one could try, e.g., Lap_n conditions based on larger protected spheres (since it seems reasonable to expect that a normative system based on small protected spheres will be most 'effective' when the agents are relatively close to each other), or generalized versions of Lap_n conditions involving three or more agents. Other ideas are to allow individual utility functions for each agent, or evolving the utility function and the normative system in parallel. In general, special treatment is required for domains such as the Explorer DALMAS where the fitness evaluations are 'noisy', i.e., subject to some degree of randomness. To deal with noisy fitness evaluations, a number of techniques are available, for example increasing the population size, and resampling and averaging the fitness. [6, Sect. 3.3] As described in Sect. 2, a variant of the latter technique is used in the Explorer DALMAS fitness evaluations. Another option regarding the evaluation function is to allow more or less variation regarding, e.g., grid sizes or shapes, number of agents, number of events per run and number of runs per normative system. However, large populations, in combination with expensive fitness calculations in each generation, are computationally challenging. The *moving average* approach by Di Pietro et al. can be used to reduce the number of samples needed per generation, and thus allow for running more generations in a given run-time. When a candidate is generated for the first time, its 'fitness array' is initialized with n fitness evaluations. For each new generation, the evaluation score is calculated only once, and the oldest score in the fitness array is replaced with the new score. A candidate's fitness is then the average of the evaluation scores in the fitness array.

Operational equivalence of norms It was observed in Sect. 3.1 that norms where grounds and consequences are based on the same set of descriptive conditions can become equivalent in the sense that they have exactly the same effect in the same situations. Let us investigate this further. First, recall the definitions of the C_i^a operators in Table 4, and note the following:

- $C_5^a c(X_\nu, x_{\nu+1}; x, s)$ iff
  $[c(X_\nu; s) \,\&\, \neg c(X_\nu; a(x_{\nu+1}, s))]$ or $[\neg c(X_\nu; s) \,\&\, \neg c(X_\nu; a(x_{\nu+1}, s))]$
- $C_{6\Lambda}^a c(X_\nu, x_{\nu+1}; x, s)$ iff
  $[c(X_\nu; s) \,\&\, \neg c(X_\nu; a(x_{\nu+1}, s))]$ or $[\neg c(X_\nu; s) \,\&\, c(X_\nu; a(x_{\nu+1}, s))]$
- $C_{6\Omega}^a c(X_\nu, x_{\nu+1}; x, s)$ iff
  $[c(X_\nu; s) \,\&\, c(X_\nu; a(x_{\nu+1}, s))]$ or $[\neg c(X_\nu; s) \,\&\, \neg c(X_\nu; a(x_{\nu+1}, s))]$
- $C_7^a c(X_\nu, x_{\nu+1}; x, s)$ iff
  $[c(X_\nu; s) \,\&\, c(X_\nu; a(x_{\nu+1}, s))]$ or $[\neg c(X_\nu; s) \,\&\, c(X_\nu; a(x_{\nu+1}, s))]$

Now suppose for example that $c \in C$, the set of potential grounds, but also $c \in D$, the set of descriptive conditions underlying the set of potential normative consequences. Let M_κ be a 'move operator', for example M_1, such that $\kappa \leq \nu$, and suppose that c is true of some agents $x_1, ..., x_\nu$ in a state s. We may then note the following:

(1) $C^a_{2\Omega}c(X_\nu, x_{\nu+1}; x, s)$, and $C^a_{4\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ are false whenever $c(X_\nu; s)$. Hence, the set of actions a that would be prohibited by $\langle M_\kappa c, P_{2\Omega}c\rangle$, resp. $\langle M_\kappa c, P_{4\Lambda}c\rangle$, is exactly the same as the set of actions that would be prohibited by $\langle M_\kappa c, P_1 c\rangle$, *viz.* the empty set.

(2) The second disjunct of $C^a_{6\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ and the second disjunct of $C^a_5 c(X_\nu, x_{\nu+1}; x, s)$ must be false. Hence, if $c(X_\nu; s)$, then $C^a_{6\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{2\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_5 c(X_\nu, x_{\nu+1}; x, s)$, which means that the actions that would be prohibited by $\langle M_\kappa c, P_{2\Lambda}c\rangle$, $\langle M_\kappa c, P_5 c\rangle$, and $\langle M_\kappa c, P_{6\Lambda}c\rangle$, are exactly the same.

(3) The second disjunct of $C^a_{6\Omega}c(X_\nu, x_{\nu+1}; x, s)$ and the second disjunct of $C^a_7 c(X_\nu, x_{\nu+1}; x, s)$ must be false. Hence, if $c(X_\nu; s)$, then $C^a_{6\Omega}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{4\Omega}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_7 c(X_\nu, x_{\nu+1}; x, s)$; so the actions that would be prohibited by $\langle M_\kappa c, P_{4\Omega}c\rangle$, $\langle M_\kappa c, P_{6\Omega}c\rangle$, and $\langle M_\kappa c, P_7 c\rangle$, are exactly the same.

In other words, given that $c(X_\nu)$ holds, $P_{2\Omega}$, $P_{4\Lambda}$, and P_1 are 'equally prohibitive', and the same holds for $P_{2\Lambda}$, $P_{4\Omega}$, and P_5, resp. $P_{2\Lambda}$, $P_{6\Omega}$, and P_7. Now suppose that $\neg c(X_\nu)$ holds for some agents $x_1, ..., x_\nu$ and some act a in s:

(4) Since $C^a_{2\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ and $C^a_{4\Omega}c(X_\nu, x_{\nu+1}; x, s)$ are false whenever $\neg c(X_\nu; s)$, the set of actions that would be prohibited by $\langle M_\kappa c', P_{2\Lambda}c\rangle$, resp. $\langle Mc', P_{4\Omega}c\rangle$, is exactly the same as the set of actions that would be prohibited by $\langle M_\kappa c', P_1 c\rangle$, *viz.* the empty set.

(5) The first disjunct of $C^a_{6\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ and the first disjunct of $C^a_7 c(X_\nu, x_{\nu+1}; x, s)$ must be false. Hence, if $\neg c(X_\nu; s)$, then $C^a_{6\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{4\Lambda}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_7 c(X_\nu, x_{\nu+1}; x, s)$, which means that the actions that would be prohibited by $\langle M_\kappa c', P_{4\Lambda}c\rangle$, and $\langle M_\kappa c', P_{6\Lambda}c\rangle$, and $\langle M_\kappa c', P_7 c\rangle$, are exactly the same.

(6) The first disjunct of $C^a_{6\Omega}c(X_\nu, x_{\nu+1}; x, s)$ and the first disjunct of $C^a_5 c(X_\nu, x_{\nu+1}; x, s)$ must be false. Hence, if $\neg c(X_\nu; s)$, then $C^a_{6\Omega}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{2\Omega}c(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_5 c(X_\nu, x_{\nu+1}; x, s)$; therefore, the actions that would be prohibited by $\langle M_\kappa c', P_{2\Omega}c\rangle$, $\langle M_\kappa c', P_{6\Omega}c\rangle$, and $\langle M_\kappa c', P_5 c\rangle$ are exactly the same.

Thus, if $\neg c(X_\nu)$ holds, then $P_{2\Lambda}$, $P_{4\Omega}$, and P_1 are equally prohibitive, and the same holds for $P_{4\Lambda}$, $P_{6\Lambda}$, and P_7, resp. $P_{2\Lambda}$, P_5, and $P_{6\Omega}$. It seems plausible that 'equally prohibitive' could be a suitable foundation for a notion of operational equivalence of norms. It is straightforward to generalize the above arguments to cases where $c\,R\,d$ or $c\,R\,d'$, i.e., where c implies d, resp., c implies d'.

4 Conclusion and Future Work

Concrete advice on how to use evolutionary mechanisms as part of the pre-runtime design of normative systems for problem-solving MAS were presented. The idea behind the methodology sketched here is to use a top-down approach of selecting (a subset of) the most 'efficient' norms from an evolved normative system, rather than a bottom-up approach of designing a normative system entirely from scratch. To illustrate the idea, a simple system, based on the DALMAS architecture for norm-regulated MAS was employed as part of the evaluation step

of an evolutionary algorithm. The results show that an evolutionary algorithm has the potential of being a useful tool when designing normative systems for problem-solving MAS.

Ideas for future work include trying to formalize and further investigate the notion of operational equivalence of norms, which was introduced in Sect. 3.2. Also left for future work is further validation of the suggested methodology, for example by applying the methodology in other domains in which the grounds of the norms and the consequences are based on different sets of descriptive conditions, or by further validating the evolved normative system for the Explorer DALMAS. One could experiment with different domain-specific parameters as well as evolutionary algorithm parameters, as suggested in Sect. 3.2, to see if better solutions can be found and thus gain more support for the ideas suggested here. It could be interesting to, e.g., explore normative systems of variable size and evaluation functions which impose a penalty for large normative systems, since in many cases it could be desirable to rely on a small number of 'rules of thumb' and avoid overly complex normative systems which may become expensive in terms of calculations. Investigating the possibility to design more accurate evolutionary operators, for example by exploiting the fact that certain norms are operationally equivalent in the Explorer DALMAS setting, also seems like a promising idea.

Acknowledgements The author is very grateful to Jan Odelstad and Magnus Boman for valuable ideas and suggestions, to participants of ICAART 2015 for discussions in relation to this paper, and to the organizers of the conference for the opportunity to expand the conference paper.

References

1. Alechina, N., Bassiliades, N., Dastani, M., Vos, M.D., Logan, B., Mera, S., Morris-Martin, A., Schapachnik, F.: Computational models for normative multi-agent systems. In: Andrighetto, G., Governatori, G., Noriega, P., van der Torre, L.W.N. (eds.) Normative Multi-Agent Systems. Dagstuhl Follow-Ups, Dagstuhl Follow-Ups, vol. 4, pp. 71–92. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2013), http://dx.doi.org/10.4230/DFU.Vol4.12111.71
2. Andrighetto, G., Castelfranchi, C., Mayor, E., McBreen, J., Lopez-Sanchez, M., Parsons, S.: (Social) norm dynamics. In: Andrighetto, G., Governatori, G., Noriega, P., van der Torre, L.W.N. (eds.) Normative Multi-Agent Systems. Dagstuhl Follow-Ups, Dagstuhl Follow-Ups, vol. 4, pp. 135–170. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2013), http://dx.doi.org/10.4230/DFU.Vol4.12111.135
3. Andrighetto, G., Governatori, G., Noriega, P., van der Torre, L.W. (eds.): Normative Multi-Agent Systems. Dagstuhl Follow-Ups, Dagstuhl Follow-Ups, vol. 4. Schloss Dagstuhl-Leibniz-Zentrum für Informatik GmbH, Dagstuhl, Germany (2013)
4. Balke, T., Cranefield, S., Tosto, G.D., Mahmoud, S., Paolucci, M., Savarimuthu, B.T.R., Verhagen, H.: Simulation and NorMAS. In: Andrighetto, G., Governatori, G., Noriega, P., van der Torre, L.W.N. (eds.) Normative Multi-Agent Systems.

Dagstuhl Follow-Ups, Dagstuhl Follow-Ups, vol. 4, pp. 171–189. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2013), http://dx.doi.org/10.4230/DFU.Vol4.12111.171

5. Darwen, P.: Computationally intensive and noisy tasks: co-evolutionary learning and temporal difference learning on backgammon. In: Proc. of the 2000 Congress on Evolutionary Computation, 2000. vol. 2, pp. 872–879 (2000), http://dx.doi.org/10.1109/CEC.2000.870731
6. Di Pietro, A., While, R.L., Barone, L.: Learning in robocup keepaway using evolutionary algorithms. In: GECCO. vol. 2, pp. 1065–1072 (2002), http://www.cs.bham.ac.uk/~wbl/biblio/gecco2002/RWA297.pdf
7. Hjelmblom, M.: Deontic action-logic multi-agent systems in Prolog. Tech. Rep. 30, University of Gävle, Division of Computer Science, Gävle, Sweden, Gävle, Sweden (2008), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-1475
8. Hjelmblom, M.: State transitions and normative positions within normative systems. Tech. Rep. 37, University of Gävle, Department of Industrial Development, IT and Land Management, Gävle, Sweden, Gävle, Sweden (2011), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-10595
9. Hjelmblom, M.: Norm-regulated transition system situations. In: Filipe, J., Fred, A. (eds.) Proc. of the 5th International Conference on Agents and Artificial Intelligence (ICAART 2013). pp. 109–117. ICAART 2013, SciTePress, Portugal (2013), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-13987
10. Hjelmblom, M.: Instrumentalization of norm-regulated transition system situations. In: Filipe, J., Fred, A. (eds.) Agents and Artificial Intelligence, Communications in Computer and Information Science, vol. 449, pp. 80–94. Springer, Berlin Heidelberg (2014), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-17672
11. Hjelmblom, M.: Normative positions within norm-regulated transition system situations. In: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT). vol. 3, pp. 238–245 (2014), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-17670
12. Hjelmblom, M.: Normative positions in multi-agent systems (2015), submitted to *Web Intelligence: An International Journal*. IOS Press
13. Hjelmblom, M., Odelstad, J.: jDALMAS: A Java/Prolog framework for deontic action-logic multi-agent systems. In: Håkansson, A., Nguyen, N., Hartung, R., Howlett, R., Jain, L. (eds.) Agent and Multi-Agent Systems: Technologies and Applications, Lecture Notes in Computer Science, vol. 5559, pp. 110–119. Springer, Berlin Heidelberg (2009), http://urn.kb.se/resolve?urn=urn:nbn:se:hig:diva-3963
14. Kanger, S.: Law and logic. Theoria 38(3), 105–132 (1972), http://dx.doi.org/10.1111/j.1755-2567.1972.tb00928.x
15. Lindahl, L.: Position and change: a study in law and logic. Synthese library, Synthese library, vol. 112. D. Reidel, Dordrecht, NL (1977), http://www.google.com/books?id=_QwWhOK8aY0C
16. Lindahl, L., Odelstad, J.: The theory of joining-systems. In: Gabbay, D., Horthy, J., Parent, X., van der Meyden, R., van der Torre, L. (eds.) Handbook of Deontic Logic and Normative Systems, vol. 1, chap. 9, pp. 545–634. College Publications, London (2013)
17. Luke, S., Hohn, C., Farris, J., Jackson, G., Hendler, J.: Co-evolving soccer softbot team coordination with genetic programming. In: Kitano, H. (ed.) RoboCup-97: Robot Soccer World Cup I, Lecture Notes in Computer Science, vol. 1395,

pp. 398–411. Springer, Berlin Heidelberg (1998), http://dx.doi.org/10.1007/3-540-64473-3_76
18. Nakashima, T., Takatani, M., Udo, M., Ishibuchi, H.: An evolutionary approach for strategy learning in robocup soccer. In: Systems, Man and Cybernetics, 2004 IEEE International Conference on. vol. 2, pp. 2023–2028. IEEE (2004), http://dx.doi.org/10.1109/ICSMC.2004.1399998
19. Odelstad, J.: Many-Sorted Implicative Conceptual Systems. Ph.D. thesis, Royal Institute of Technology, Stockholm, Sweden (2008), http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-9624, qC 20100901
20. Odelstad, J., Boman, M.: Algebras for agent norm-regulation. Annals of Mathematics and Artificial Intelligence 42, 141–166 (2004), http://dx.doi.org/10.1023/B:AMAI.0000034525.49481.4a
21. Verhagen, H., Boman, M.: Norms can replace plans. In: IJCAI'99 Workshop on Adjustable, Autonomous Systems (1999), ftp://ftp.dsv.su.se/users/verhagen/IJCAI99WS.rtf
22. Whitley, D.: An overview of evolutionary algorithms: practical issues and common pitfalls. Information and Software Technology 43(14), 817–831 (2001), http://dx.doi.org/10.1016/S0950-5849(01)00188-4