Agents, Norms and Forest Cleaning

Jan Odelstad

Department of Mathematics, Natural and Computer Sciences, University of Gävle,
SE-801 76 Gävle, Sweden,
jod@hig.se

Abstract. The automation of forest cleaning presupposes principles for choosing those trees that ought to be taken away and those that shall be left standing. In this paper, which is a report on a work in progress, the question is raised whether those principles can be structured as a combination of a normative system and a utility function. Of special interest is the possibility that the agent system can evaluate the efficiency of the normative system and the utility function and, furthermore, suggest improvements of them. Earlier works on norms and norm-regulation of agent systems that the author has been involved in are used to elucidate the problem area discussed in the paper.

1 Introduction

Within economic theory the consumer's behaviour has traditionally been described as determined by a utility function. During the latest three decades there has been a growing interest among researchers in how norms (for example rules of law) give restrictions on the behaviour induced by the utility function.[1] The behaviour of the consumers or other economic agents, according to this model, is the result of interplay between optimization of the utility function and restrictions due to norms. We may perhaps speak of *norm-regulated Homo oeconomicus*. It has also been suggested that a model of this kind could be used for regulating the behaviour of artificial agents. We can perhaps call this model *Agent oeconomicus norma.* The role that norms will have in regulating the behavior of agents is, according to this model, to delimit the autonomy of the agents. Metaphorically one can say that the norms define the scope (*Spielraum*) for an agent. The agent chooses the act it likes best within the scope determined by the norms.

In this preliminary report on a work in progress, I will discuss some aspects of how norms can be used to regulate the behaviour of multiagent-systems. I will do this at least partially with a concrete problem area in view, namely the automation of forest management treatments, especially the cleaning of young forest stands. The automation of forest cleaning presupposes principles for choosing, in a state of incomplete information, those trees that ought to be taken away

[1] Cf. for example [4] p. 518, where the so called Coase theorem is discussed.

Dagstuhl Seminar Proceedings 07122
Normative Multi-agent Systems
http://drops.dagstuhl.de/opus/volltexte/2007/917

and those that shall be left standing. I will discuss the possibility of formulating those principles as a kind of norms and in that way partially regulate the cleaning process by a normative system.

Norm-regulation of agents presupposes a precise and significant representation of norms and normative systems. A norm is here represented as an implicative sentence where the antecedent is a descriptive condition stating the circumstances of an agent, and the consequent is a condition expressing the normative or deontic position that the agent has with respect to a state of affairs. Hence, from the norms of the system will follow a deontic structure over possible state of affairs implying that some states may be permissible while the rest are non-permissible. The "wish" or "desire" of an agent is represented as a preference structure over possible states or situations. The agent chooses an act which leads to a permissible state it prefers the most.

In [14] the ideas outlined above were developed using the typology of normative (deontic) positions developed by Kanger and Lindahl and the algebraic representation of normative systems developed by Lindahl and myself. One of the results in [14] was a scheme for how normative positions will restrict the set of actions that the agents are permitted to choose from. The possibility of using the proposed theory in the construction of abstract architectures for multiagent-systems was discussed. The method used for describing abstract architectures was based on the definition of set-theoretical predicates.

Thee is a need for revisions and developments of [14] in different aspects. The algebraic representation of normative systems has been further developed since [14] was published, for example by the use of intermediaries, especially open intermediaries. The application of the Kanger-Lindahl typology of normative positions can be a more complex task than indicated in [14], for example two-agent types of normative positions can be used in addition to just one-agent-types. Furthermore, the characterization of a norm-regulated multiagent-system can be made more flexible. The intended extension and revision of [14] will be tested by an application to the decision-making problem of an idealized forest cleaning agent.

2 Automation of forest cleaning[2]

2.1 Cleaning *in silico*

In forest industry there is an increasing interest in the automation of forest management treatments, perhaps with the ultimate goal that autonomous robots will be able to do a substantial part of such work. But before robots of this kind can be constructed a lot of difficult problems must be solved, for example how the robots will perceive the environment and how they will transport itself. But there are also decision-making problems involved. Three important kinds of forest management treatments are cleaning, thinning and harvesting, and they all require methods or principles for making decisions of which trees shall be

[2] This section is based on [2].

taken away (removed) and which will be left standing. Such "remove-decisions" must be made on-line with information only of the robot's neighbourhood and about that part of the stand already cleared. The treatment cannot be evaluated until the actual stand is completely cleared. To test and evaluate principles for remove-decisions by real-world-experiments (field experiments) is expensive and time-consuming. It is therefore an interesting question wether evaluating experiments could be made *in silico*, i.e. through simulation.

In [1] a platform for simulation of young forest stands is presented. Given field data of a special type of young forest, for example a 10 years old somewhat damp spruce forest in the middle of Sweden on 200 m above sea level, it is possible to simulate different stands of this type of forest. Field data of a few different types of young forests has so far been used for simulation. As a base for the simulation of different stands of the same forest type, it is of course also possible to use man-made, artificial data, or to assign values to the parameters that govern the simulation.

One of the goal with our present work on automation of forest cleaning is to formulate different principles for making the remove-decisions, test the principles in simulated forests of different types and evaluate and compare the results. We are especially interested in the possibility that, given a method for evaluating the result of cleaning, the system can improve the decision-making principles and even suggest new ones on the bases of machine learning.

How the principles for the remove-decisions ought to be formally represented seems to be a complicated question. One possibility we want to investigate is to formulate the principles as a normative system supplemented by a utility function, i.e. if the automation of forest cleaning could be modelled as an *Agent oeconomicus norma.*

Forest cleaning is a kind of activity and to elucidate some formal aspects of norm-regulation of forest cleaning, the next subsections are devoted to some general remarks on the structure of an activity and its evaluation.

2.2 The formal structure of an activity

An activity is based on actions performed by one or more agents. Accordingly, one can distinguish between one-agent-activities and multi-agent-activities. The execution of a specific instance of an activity involves the performance of a number of actions, often appropriately represented as a sequence of actions. Let V be an activity and c an instance of V. The execution of the instance c of the activity V starts from an input and results in an output. Let c^i be the input of c and c^o the output of c. The input of c is often an initial state while the output of c is often a set of possible final states. If the output of an instance of the activity V is always singleton, then the execution of V is deterministic. If c is an instance of a deterministic activity V and $c^o = \{s\}$ we will often, for the sake of simplicity, say that $c^o = s$, i.e. we identify the output with the element in the singleton set.

Suppose that V is an activity. There may be different ways of (or different procedures for) executing instances of the activity V and each way or procedure

is represented by a function F such that $F(c^i) = c^o$. If F and G are different ways of executing V we say that F and G are extensionally equivalent, which is denoted $F =_e G$, if $F(c^i) = G(c^i)$ for all instances c of V.

There are different modes of definition of F and depending on the mode of definition there are different kinds of questions to ask. We will here give three examples.

(I) F is determined as the agent ω's way of executing the activity V. This presupposes that the agent is sufficiently "reliable" such that one can suppose that the agent has a procedure for executing the activity V. In this case one can be interested in
   (i) making explicit the rules that ω uses, such that every competent agent can determine the value of F for an argument c^i by applying the rules, or
   (ii) determining a computational function G which is extensionally equivalent to F.

(II) F is defined by a system of rules such that an agent system A which complies with the rules "computes" F. In this case one may want to characterize F as a computationally more effective function that is not necessarily agent-based.

(III) F is defined as a computational function, in which case it can be of interest to determine a system of rules such that if an agent follows the rule the result will be $F(c^i)$ for all instances c of V.

2.3 Forest cleaning as a kind of activity

It does not seem to be adequate to consider 'forest cleaning' as an activity - it is too broad a concept and is more properly classified as a kind of activity. However, the cleaning of a special kind or type of forest for a certain aim or purpose is an activity. An instance of a cleaning activity is the cleaning (specified in an appropriate way) of a stand. Input of the instance is the stand before cleaning and the output is (a) the stand after cleaning if the activity is deterministic and (b) the set of possible results of cleaning the stand if the cleaning activity is indeterministic. A cleaning activity is often executed by one agent alone and cleaning is therefore in many contexts a one-agent-activity.

Let $\mathfrak{R}$ denote the activity 'cleaning of the kind κ of forests for the purpose π'. κ can, for example, be a 'mixed forest' or a 'deciduous forest'. π can be related to the intended use of the forest in the future. An instance c of this activity is the cleaning of a certain stand p. The input of the instance, i.e. c^i, is the stand p before it is cleaned and we denote it p_0. This means that $c^i = p_0$. The output of c is the stand p after the cleaning has been done. Agents executing $\mathfrak{R}$ are today human beings. A human cleaner who is going to execute the instance c does not need a description of p_0, but an artificial cleaner needs a description or representation of the input of c. Such a representation may consist, for example, of a description of each tree in the stand, the position of the trees, the topography of the land, the quality of the soil, the existence of obstacles (for example large stones and ditches) and the climate (micro as well as large scale) of the stand.

For testing different ways of executing cleaning, it is convenient to use a platform for simulations of young forest stands and such a platform is described in [7] and [1]. The platform is used for two different tasks, (a) to create replicate forest stands and (b) to perform cleaning on field data or on simulated data. The objective behind the first task is to create replicates of forest stands that belong to certain forest types. The basic data for a replicate simulation can be field data or simulated data. Data of a given forest type can be used for constructing a decision tree structure and probabilities are calculated for different parameter values of a tree, such as diameter, eventual damage and species. The decision tree structure can then be used for creating replicate forest stands that are different but represent the same forest type as the stand form which the data were collected. A user can also apply her own principles of forest stand parameters and thus decide more or less strictly how the resulting forest is going to be.

In [17], Vestlund et.al. describe a way of cleaning young forest stands as an on-line algorithm expressed in an informal language. The algorithm is at least partially based on interviews with professional cleaners. In [17] there is also an implementation of the algorithm in a programming language and cleaning *in silico* of forest stands represented by computer files are executed. In [2] results of cleaning simulated stands using principles based on the work by Vestlund et.al. are presented.

2.4 Evaluating activity procedures

Different procedures for executing an activity V can be more or less good or appropriate. It is therefore important in many contexts to evaluate different ways of executing activities. Suppose that F and G are different ways of executing V. It is important to note the difference between how good F is as a way of executing a given instance of V and how good F is a way of executing V in general. It is thus important to distinguish between

1. $F(c^i) \succ^V G(c^i)$, i.e. the execution of the instance c of V by F is better than by G.
2. $F \succ_V G$, i.e. F is quite generally a better way of executing V than G.

In certain cases there exists a measure u_V of how good different procedures for executing an instance c of the activity V is and a measure U_V of how good different procedures for the activity V is generally. It is reasonable to suppose that the following holds:

(a) $u^V(F(c^i)) > u^V(G(c^i))$ iff $F(c^i) >^V G(c^i)$
(b) $u_V(F) > u_V(G)$ iff $F >_V G$.

If for all instances c of an activity V it holds that $F(c^i) >^V G(c^i)$, then it seems uncontroversial to conclude that $F >_V G$. But if for some instances c it holds that $F(c^i) >^V G(c^i)$while for other instances c it holds that $G(c^i) >^V F(c^i)$ then it is more difficult, in many cases impossible, to decide if $F >_V G$ or not.

The binary relations $>^V$ and $>_V$ and the measures u^V and u_V concern how good different execution procedures are (for given instances or generally) "all things considered". These relations and measures can be difficult to establish since appropriateness ("goodness"), all things considered, is in many contexts a very complex attribute. However, appropriateness with respect to a certain aspect (attribute) α may be easier to ascertain. Let

- $F(c^i) >_\alpha^V G(c^i)$ denote that F is a better way with respect to α than G to execute the instance c of V; if the activity is cleaning then α can be for example the number of trees per unit area or the average distance between the trees.
- $F >_{V,\alpha} G$, denote that F is a better way with respect to α than G to execute V.
- $u_\alpha^V(F(c^i))$ is a measure of how good F is with respect to α as a way of executing the instance c of the activity V.
- $u_{V,\alpha}(F)$ is a measure of how good F is generally with respect to α as a way of executing the activity V.

u_α^V and $u_{V,\alpha}$ can be considered as a kind of utility function. Utility functions are largely used in economics. In addition, utility functions play an important role in the study of intelligent agents within the discipline of artificial intelligence, as the following quotation from [16] p. 52 emphasizes: "... when there are conflicting goals, only some of which can be achieved ..., the utility function specifies the appropriate tradeoffs. ... when there are several goals that the agent can aim for, none of which can be achieved with certainty, utility provides a way in which the likelihood of success can be weighed up against the importance of the goals."

Suppose that different procedures for executing V is evaluated with respect to the attributes $\alpha_1,\ldots,\alpha_k$. Then

$$u^V\left(F\left(c^i\right)\right) = f\left(u_{\alpha_1}^V(F(c^i),...,u_{\alpha_k}^V(F(c^i))\right).$$

The form of the function f can differ considerably in different contexts. But it is often a desideratum that f has a simple form, for example being additive or multiplicative.

For many aspects α of importance for how good F is as a way of executing V, but certainly not for all aspects, it seems reasonable that the following holds:

- $F(c^i) >_\alpha^V G(c^i) \Rightarrow F(c^i) >^V G(c^i)$ ceteris paribus (everything else held constant)
- $F >_{V,\alpha} G \Rightarrow F >_V G$ ceteris paribus (everything else held constant).

It is of course desirable to have a complete method for evaluating different ways of executing an activity but when this is not possible the concepts introduced above can still make a partial evaluation possible, and this may in certain situations be of great interest.

Suppose that the relation $>_V$ is defined for the activity V. Then an optimal way of executing V is a procedure F that is maximal with respect to $>_V$. It

is usually desirable to choose an optimal way of executing V, but it can be difficult to find such a way if the number of different ways to execute V is large. The search for an optimal way of executing V may then be replaced by a more restricted goal, viz. the search for an acceptable way, i.e. a way that is good enough (sufficiently acceptable).

3 Norm-regulation of cleaning agents[3]

In [14], an abstract architecture for idealized multi-agent systems, whose behaviour is regulated by normative systems, is developed. The idea behind the architecture is roughly the following:

"When it is agent ω's turn to move it chooses an act out of a set of feasible alternatives and the result will be that the system enters a new state; which state depends on the actual state of the system when the act is performed. The agent's choice is determined partially by the preference ordering of the possible states and partially by the deontic structure: the agent chooses that act which leads to the best outcome of all permissible actions. If an action is permissible or not depends on whether the result of performing the action leads to a state which satisfies a condition which is forbidden according to the normative system regulating the multi-agent system." ([14] p. 152).

The architecture is articulated as a definition of a deontic action-logic based multiagent-system. Since 'MAS' is the standard abbreviation of 'multiagent-system', we call the kind of system discussed here 'DALMAS'.

In this section the possibility to use DALMAS as the architecture for a cleaning agent is outlined. At this preliminary stage, a cleaning agent is regarded as "a solitary being" and, hence, a cleaning DALMAS is a one-agent-system (thus more correctly a Daloas), but we will here regard a one-agent-system as a degenerated MAS. But at a later stage, more then one agent may be involved, for example can "nature" be regarded as an agent or the individual trees be regarded as agents. The last mentioned alternative is especially interesting if the growth of a forest stand is incorporated in the simulation.

The formal definition of a DALMAS is summarized in the next section. In this section is given an outline of how the definition can be applied in the context of cleaning. Let us consider the cleaning of a stand p.

The stand p is divided into n different areas. A state is the stand with i areas cleaned, where $1 \leq i \leq n$, and a specification of what area to clean next. The initial state is the stand with 0 areas cleaned and the final state is the state with n areas cleaned. We let each area be denoted by a unique number between 1 and n. Let S_i be the i:th state. Let C_i be the set of cleaned areas and U_i the set of uncleaned areas in S_i. Thus, $C_i \cup U_i = \{1, 2, \ldots, n\}$ and $C_i \cap U_i = \emptyset$. C_i contains i numbers and U_i $n - i$ numbers. $S_i = \langle C_i, U_i, j\rangle$ where j is the area which will be cleaned next, i.e. $j \in U_i$ and $S_{i+1} = \langle C_i \cup \{j\}, U_i \backslash \{j\}, k\rangle$ for some $k \in U_i \backslash \{j\}$.

[3] This section is based on [1].

The norm of a cleaning DALMAS is expressed in a conditional sentence of the following form, where DEOP is a deontic operator, for example 'it shall be the case that', 'it may be the case that ' or 'it may not be the case that':

Given that the actual state of the system has the property P, then DEOP(the next state of the system has the property Q).
Let us give a few examples of sentences that have the right form for being norms (which of course does not imply that they are norms):

(a) If there is only one undamaged tree in the area to be cleaned with diameter within the desirable range, then this tree shall be saved.
(b) If there is at least one undamaged tree in the area to be cleaned with diameter within the desirable range, then a damaged tree with diameter below the desirable range may be taken away.
(c) If, in the area to be cleaned, a tree t is damaged and is closer than 0.5m to an undamaged tree with diameter within the desirable range and with distances to other undamaged trees larger than 0.5m, then t may not be saved.

The antecedent of a norm is thus a descriptive sentence and the consequent is a deontic sentence. In the definition of DALMAS, the Kanger-Lindahl theory of normative positions is used to obtain a logically powerful framework for expressing deontic sentences.

In many situations, the norms for a DALMAS do not determine the action to be taken in each state, but utility considerations are also necessary. If we have the relation $>_{\mathfrak{R}}$ at hand, where $\mathfrak{R}$ denotes the activity 'cleaning of the kind κ of forests for the purpose π', then we can search for the optimal way of cleaning the actual area, given the condition that the way of cleaning satisfies the given norms.

4 The definition of a Dalmas

The aim of [14] was to present a theory of how norms can be used to regulate the behaviour of multiagent-systems on the assumption that the role of norms is to define the *Spielraum* for an agent. The theory can be summarized as follows. The norms for a MAS are regarded as belonging to a normative system and such a system is represented algebraically as a Boolean joining system containing a Boolean quasi-ordering of grounds and a Boolean quasi-ordering of consequences. The norms are joinings from the Boolean quasi-ordering of grounds to the Boolean quasi-ordering of consequences, and the specific normative content of a normative system is given by the set of minimal norms. The consequences are expressed using operators on conditions corresponding to the Kanger-Lindahl-types of one-agent-positions. An important step in the theory construction was the specification under what circumstances the sentence $T_i d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ implies that an action a is prohibited for the agent ω in the state s. (See section 6.) An action a was regarded as a function, which is the usual way of representing an action in decision theory, and d is a ν-ary condition on agents, true or

false in the situation s. An abstract architecture based on the theory of norm-regulation of behaviour was defined, and a MAS having this architecture was called a norm-regulated DALMAS.

DALMAS is a global clock (synchronous update), global state, global dynamics system. It can be viewed as a simplification constructed for conceptual and computer simulation purposes. In particular, DALMAS can be used as a model system for studying the interplay between preferences and norms in MAS architectures.

A DALMAS is an ordered 7-tuple $\langle \Omega, S, A, \mathcal{A}, \Delta, \Pi, \Gamma \rangle$ containing

- an agent set Ω ($\omega, \varkappa, \omega_1, ...$ elements in Ω),
- a state or phase space S ($r, s, s_1, ...$ elements in S),
- an action set A such that for all $a \in A$, $a : \Omega \times S \longrightarrow S$ such that $a(\omega, r) = s$ means that if the agent ω performs the act a in state r, then the result will be state s ($a, b, a_1, ...$ elements in A),
- a function $\mathcal{A} : \Omega \times S \longrightarrow \wp(A)$ where $\wp(A)$ is the power set of A; $\mathcal{A}(\omega, s)$ is the set of acts accessible (feasible) for agent ω in state s,
- a deontic structure-operator $\Delta : \Omega \times S \longrightarrow \mathcal{D}$ where $\mathcal{D}$ is a set of deontic structures of the same type with subsets of A as domains and $\Delta(\omega, s)$ is ω's deontic structure on $\mathcal{A}(\omega, s)$ in state s,
- a preference structure-operator $\Pi : \Omega \times S \longrightarrow \mathcal{P}$ where $\mathcal{P}$ is a set of preference structures of the same type with subsets of A as domains and $\Pi(\omega, s)$ is ω's preference structure on $\mathcal{A}(\omega, s)$ in state s,
- a choice-set function $\Gamma : \Omega \times S \longrightarrow \wp(A)$ where $\Gamma(\omega, s)$ is the set of actions for ω to choose from in state s.

Note that in the definition the Cartesian product $\Omega \times S$ motivates the introduction of a name for the elements in $\Omega \times S$. Let $\mathfrak{D}$ be a DALMAS. A *situation* for the system $\mathfrak{D}$ is determined by the agent to move ω and the state s. A situation is represented by an ordered pair $\langle \omega, s \rangle$. The set of situations for $\mathfrak{D}$ is thus $\Omega \times S$.

In a DALMAS, all the agents have the same initial set of actions. The set of actions to choose from (the choice-set) in a situation $\langle \omega, s \rangle$ is determined by the agent's deontic structure $\Delta(\omega, s)$ and preference structure $\Pi(\omega, s)$. If $\Gamma(\omega, s)$ consists of one action, then this action applied in the situation $\langle \omega, s \rangle$, i.e. $[\Gamma(\omega, s)]\,(\omega, s)$, is the resulting state when ω acts in state s.

A *simple* DALMAS is a DALMAS containing the following simple versions of Δ, Π and Γ.

1. $\Delta(\omega, s) \subseteq \mathcal{A}(\omega, s)$ and $\Delta(\omega, s)$ is the set of permissible actions for ω in the state s,
2. $\Pi(\omega, s) = \langle \mathcal{A}(\omega, s), \succsim \rangle$ where $\succsim$ is a weak ordering
3. $\Gamma(\omega, s) = \{x \in \Delta(\omega, s) : \text{ for all } y \in \Delta(\omega, s),\ x \succsim y\}$.

Hence, in a simple DALMAS the choice-set consists of the best actions which are permissible. Among the elements in A there can be a pass action, which

means the agent does nothing. If we combine the existence of such an action with very short clock cycles, we obtain systems with close to asynchronous behaviour.

A DALMAS is not deterministic, since it does not determine in which order the agents are going to move, and the choice-set may contain more than one action in every situation. Let us therefore make the following definition.

Definition 1. *A* deterministic DALMAS *is an ordered 9-tuple* $\langle \Omega, A, S, \mathcal{A}, \Delta, \Pi, \Gamma, \tau, \gamma \rangle$ *such that* $\langle \Omega, A, S, \mathcal{A}, \Delta, \Pi, \Gamma \rangle$ *is a* DALMAS *and* $\tau : \Omega \longrightarrow \Omega$, $\gamma : \wp(A) \longrightarrow A$.

The intended interpretation is the following:

- $\tau : \Omega \longrightarrow \Omega$ is a turn-operator such that $\tau(\omega) = \varkappa$ means that it is $\varkappa$'s turn after ω; τ determines a simple agent priority.
- $\gamma : \wp(A) \longrightarrow A$ is a tie-breaking function, determining which of several permissible and equally preferred actions to choose.

An important tool in the present study is the characterization of abstract architectures by the definitions of set-theoretical predicates. Among the abstract architectures defined in this way, the most important one is a norm-regulated DALMAS. This is just the first step towards a theory of architectures for MAS that restricts the behaviour of the multiagent-system using norms. The theory can be developed by defining a number of set-theoretical predicates that are specifications of the predicate DALMAS, and we can obtain a hierarchy of predicates with DALMAS as its root.

5 The algebraic representations of norms and normative systems

The method used for representing norms in an architecture for norm-regulated MAS can be of importance for the effectiveness of the architecture. Let me mention a few examples of what can be regarded as desiderata for a norm-representation method.

1. The system of norms is depicted in a lucid, concise and effective way.
2. Changes and extensions of the normative system are easily described.
3. The normative system can be divided in different parts which can be changed independently.
4. The multi-agent system can by itself change the normative system wholly or partially.

The last item in the list may deserve a comment. It is often difficult to predict the effect of a normative system for a MAS or the effect of a change of norms. It is therefore desirable that the MAS can by itself evaluate the effect of the normative system and compare the result with other normative systems that it changes to. The result can be a kind of evolution of normative systems obtained by machine learning.

In a series of papers (among which are [15], [9], [10], [11] and [12]) Lars Lindahl and I have developed an algebraic approach to the study of normative systems. One of our main tools in this endeavour is the theory of a Boolean quasi-ordering, which is an extension of the theory of Boolean algebras. A norm is regarded as consisting of two objects, a ground and a consequence standing in a relation to each other. The ground belongs to one Boolean quasi-ordering and the consequence to another. Therefore, we can view a normative system as a set of joinings of a Boolean quasi-ordering of grounds to a Boolean quasi-ordering of consequences. A normative system $\mathcal{S}$ can therefore be represented as a Boolean joining system $\langle \mathcal{B}_1, \mathcal{B}_2, J \rangle$ where $\mathcal{B}_1$ is a Boolean quasi-ordering of ground-conditions, $\mathcal{B}_2$ a Boolean quasi-ordering of consequence conditions and the set J of norms where $J \subseteq B_1 \times B_2$. One can define a qu asi-ordering $\trianglelefteq$ expressing how narrow norms are and determine the set of minimal elements of J, $\min J$, with respect to $\trianglelefteq$. The set $\min J$ characterizes J in the following way:

$$\langle a_1, a_2 \rangle \in J \text{ iff } \exists \langle b_1, b_2 \rangle \in \min J : \langle b_1, b_2 \rangle \trianglelefteq \langle a_1, a_2 \rangle .$$

Given certain general presuppositions, one can choose a subset C of $\min J$ from which $\min J$ can be inferred and which therefore also determines J. We call such a set C for a base of minimal elements of J. In many contexts the elements in C can be represented by intermediaries (intermediate concepts). Intermediaries are determined by the condition that constitute its ground and the condition that constitutes its consequences.

Within the algebraic representation of norms the consequences are normative conditions. In [14] simple normative conditions were constructed by letting a "normative position operator" T_i, $1 \leq i \leq 7$, operate on descriptive conditions. Compound conditions are Boolean combinations of simple conditions. The operators $T_1, ..., T_7$ are applications of Lindahl's one-agent-types of normative positions, which are briefly described in the next section.

6 The Kanger-Lindahl typology of normative positions

Based on the work by Stig Kanger ([5], [6]), Lars Lindahl developed three systems of types of normative positions, see [8]. The simplest one is the system of one-agent types of normative position, and in [14] we made only use of this system. This kind of types is constructed in the following way. Let $\pm\alpha$ stand for either of α or $\neg\alpha$. Starting from the scheme $\pm\text{May}\pm\text{Do}(x, \pm q)$, where $\pm$ stands for the two alternatives of affirmation or negation, a list is made of all maximal and consistent conjunctions, 'maxiconjunctions', such that each conjunct satisfies the scheme.[4] Maximality means that if we add any further conjunct, satisfying the scheme, then this new conjunct either is inconsistent with the original conjunction or redundant. Note that the expression $\neg\text{Do}(x,q)\& \neg\text{Do}(x, \neg q)$ expresses x's passivity with regard to q. Here this expression is abbreviated as $\text{Pass}(x, q)$. By this procedure the following list of seven maxiconjunctions is obtained, which are denoted $\mathbf{T}_1(x, q)$,...,$\mathbf{T}_7(x, q)$ (see [8] p. 92).

[4] The notion of 'maxiconjunction' was introduced in Makinson (1986) p. 405f.

$\mathbf{T}_1(x,q)$: MayDo(x,q) & MayPass(x,q) & MayDo$(x,\neg q)$.
$\mathbf{T}_2(x,q)$: MayDo(x,q) & MayPass(x,q) & $\neg$MayDo$(x,\neg q)$
$\mathbf{T}_3(x,q)$: MayDo(x,q) & $\neg$MayPass(x,q) & MayDo$(x,\neg q)$.
$\mathbf{T}_4(x,q)$: $\neg$MayDo(x,q) & MayPass(x,q) & MayDo$(x,\neg q)$.
$\mathbf{T}_5(x,q)$: MayDo(x,q) & $\neg$MayPass(x,q) & $\neg$MayDo$(x,\neg q)$.
$\mathbf{T}_6(x,q)$: $\neg$MayDo(x,q) & MayPass(x,q) & $\neg$MayDo$(x,\neg q)$
$\mathbf{T}_7(x,q)$: $\neg$MayDo(x,q) & $\neg$MayPass(x,q) & MayDo$(x,\neg q)$.

$\mathbf{T}_1$,...,$\mathbf{T}_7$ are called the types of one-agent positions. Given the underlying logic, the one-agent types are mutually disjoint and their union is exhaustive. Note that $\neg$MayDo (x,q) & $\neg$ MayPass (x,q) & $\neg$MayDo$(x,\neg q)$ is logically false, according to the logic of Shall and May.

The ground of a norm is usually a descriptive condition, while the consequence is a deontic condition. In [10] we use the one-agent-types in the Kanger-Lindahl theory of normative positions as operators on descriptive conditions to get deontic conditions. As a simple example, suppose that r is a unary condition. Then $T_i r$ (with $1 \leq i \leq 7$) is the binary condition such that

$$T_i r(y,x) \text{ iff } \mathbf{T}_i(x, r(y)),$$

where $\mathbf{T}_i(x,r(y))$ is the i:th formula of one-agent normative positions. Note that for example $\mathbf{T}_3(x,r(y))$ means

$$\text{MayDo}(x,r(y)) \;\&\; \neg\text{MayPass}(x,r(y)) \;\&\; \text{MayDo}(x,\neg r(y)).$$

If $\langle p, T_i r\rangle$ is a norm, then from $p(x_1,x_2)$ we can, by using the norm, infer $T_i r(x_1,x_2)$ and thus also $\mathbf{T}_i(x_2,r(x_1))$, which means that, with regard to the state of affairs $r(x_1)$, x_2 has a normative position of type $\mathbf{T}_i$.

7 Normative positions regulating actions

The idea behind a norm-regulated DALMAS is that the deontic structure operator Δ is defined in terms of a normative system $\mathcal{S}$ in the sense that what is permissible to do in a situation is determined by $\mathcal{S}$. This idea can be explicated in the following way. Let

$T_i d(\omega_1,...,\omega_\nu,\omega;\omega,s)$ mean that in the situation where it is ω's turn to draw and the state of the system is s, ω has the normative position of type T_i with regard to the state of affairs $d(\omega_1,...,\omega_\nu)$.

Prohibited$_{\omega,s}(a)$ mean that in the situation where it is ω's turn to draw and the state of the system is s, ω is prohibited to execute the act a.

The following seven principles establish connections between the condition $T_i d$ and the predicate Prohibited (see [14] p. 160f.):

1. From $T_1 d(\omega_1,...,\omega_\nu,\omega;\omega,s)$ follows no restriction on the acts.
2. From $T_2 d(\omega_1,...,\omega_\nu,\omega;\omega,s)$ follows that
   if $d(\omega_1,...,\omega_\nu;s)$ and $\neg d(\omega_1,...,\omega_\nu;a(\omega,s))$ then Prohibited$_{\omega,s}(a)$.

3. From $T_3 d(\omega_1, ..., \omega_\nu \omega; \omega, s)$ follows that
   if $[d(\omega_1, ..., \omega_\nu; s)$ iff $d(\omega_1, ..., \omega_\nu; a(\omega, s))]$ then Prohibited$_{\omega,s}(a)$.
4. From $T_4 d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ follows that
   if $\neg d(\omega_1, ..., \omega_\nu; s)$ and $d(\omega_1, ..., \omega_\nu; a(\omega, s))$ then Prohibited$_{\omega,s}(a)$.
5. From $T_5 d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ follows that
   if $\neg d(\omega_1, ..., \omega_\nu; a(\omega, s))$ then Prohibited$_{\omega,s}(a)$.
6. From $T_6 d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ follows that
   if not $[d(\omega_1, ..., \omega_\nu; s)$ iff $d(\omega_1, ..., \omega_\nu; a(\omega, s)]$ then Prohibited$_{\omega,s}(a)$.
7. From $T_7 d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ follows that
   if $d(\omega_1, ..., \omega_\nu; a(\omega, s))$ then Prohibited$_{\omega,s}(a)$.

These principles can be used to define the deontic structure-operator Δ. One possibility is to let $\Delta(\omega, s)$ be the set of feasible acts a that are not eliminated as Prohibited$_{\omega,s}(a)$ according to the rules (1)-(7) above, where

Prohibited$_{\omega,s}(a)$ is equivalent to $\neg$Permissible$_{\omega,s}(a)$.

Hence,

$\Delta(\omega, s) = \{\text{Permissible}_{\omega,s}(a) : a \in A\}$.

Note that in the outset all feasible acts are permissible, i.e. for all $a \in A$ Permissible$_{\omega,s}(a)$. The basic idea is that we eliminate elements from the set of permissible acts for ω in s using the norms and sentences expressing what holds for the agents with respect to grounds in the norms. To facilitate the presentation it is convenient to introduce the following six operators on state-conditions:

$E_2^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff $[d(\omega_1, ..., \omega_\nu; s)$ and $\neg d(\omega_1, ..., \omega_\nu; a(\omega, s))]$
$E_3^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff $[d(\omega_1, ..., \omega_\nu; s)$ iff $d(\omega_1, ..., \omega_\nu; a(\omega, s))]$
$E_4^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff $[\neg d(\omega_1, ..., \omega_\nu; s)$ and $d(\omega_1, ..., \omega_\nu; a(\omega, s))]$
$E_5^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff $[\neg d(\omega_1, ..., \omega_\nu; a(\omega, s))]$
$E_6^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff not $[d(\omega_1, ..., \omega_\nu; s)$ iff $d(\omega_1, ..., \omega_\nu; a(\omega, s)]$
$E_7^a d(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ iff $d(\omega_1, ..., \omega_\nu; a(\omega, s))$.

Note that for all i, $2 \leq i \leq 7$, $(T_i d \wedge E_i^a d)(\omega_1, ..., \omega_\nu, \omega; \omega, s)$ implies that Prohibited$_{\omega,s}(a)$

In [3] an implementation in Prolog of the theory of a norm-regulated DALMAS is presented. The algebraic model for the DALMAS is inspected and instrumentalized through an executable logic program. In particular, important issues in the transition from a set-theoretical description to a Prolog implementation are discussed. Results include a general-level Prolog implementation, which may be freely used to implement specific systems.

According to the definition of a DALMAS, an agent "enters" a set of normative positions when it is its turn to move, and this set is determined by the normative system and the actual state. Normative positions can play a role in the abstract architectures for agent systems in other ways as well. One possibility is the following. A state s contains a description of all the normative positions for all agents in that state. Of course, the normative positions for an agent can be

changed when another agent moves, but an agent can only execute its rights or fulfill its duties when it is its turn to move. How the normative positions change when an agent moves is determined by the normative system for the DALMAS. (Some of the normative positions for an agent in a state can be conditional, so that they can be executed only when certain requirements are satisfied.)

8 Concluding remarks

Forest cleaning is a kind of activity and in the paper some aspects of the formal structures and the evaluation of activities have been discussed. The automation of forest cleaning presupposes, inter alia, principles for making "remove-decisions" and the question has been raised if such principles can be formulated as a combination of a normative system and a utility function. Some results about the representation of normative systems and the norm-regulation of multiagent-systems that may be used in the investigation of the question at issue has been outlined.

For the possibility of using norms in the automation of forest cleaning in the way outlined above, it may be an important issue whether the cleaning system can optimize the system of norms regulating its remove-decisions. This is a special case of a more general problem: Suppose that $\mathfrak{D}$ is a DALMAS, where the agents cooperate to solve a problem. Which normative system will lead to the most effective behavior of the system? Of course, it is desirable that $\mathfrak{D}$ itself could determine the optimal normative system for the task in question. Given a set of grounds and a set of consequences, which together constitute the vocabulary of the system, $\mathfrak{D}$ can test all possible sets of minimal norms (in some cases satisfying certain constraints). If there is a function for evaluating the result of a run of $\mathfrak{D}$, then different normative systems can be compared and the best system can be chosen. A change of vocabulary corresponds to a "mutation" among normative systems and can lead to dramatic changes in the effectiveness. Note that, in principle, the evaluation function can be very complicated, for example it can be multi-dimensional.

Acknowledgement

I am very grateful to my colleagues involved in different aspects of this work: Ulla Ahonen-Jonnarth, Magnus Blom, Magnus Boman and Lars Lindahl. Financial support was given by the University of Gävle.

References

1. Ahonen-Jonnarth U, Odelstad J. (2005). Simulation of cleaning of young forest stands. *Reports from Creativ Media Lab,* University of Gävle, Report 2005:2, 25 pp.
2. Ahonen-Jonnarth, U. & Odelstad, J. (2006). Evaluation of Simulations with Conflicting Goals with Application to Cleaning of Young Forest Stands. *Proceedings of ISC 2006* (Fourth Annual International Industrial Simulation Conference), Palermo, Italy, June 5-7, 2006.

3. Blom, M. (2007). *Deontic Action-Logic Multi-Agent Systems in Prolog.* Master's Thesis, Department of Information Technology, Uppsala University.
4. Gravelle, H. & Rees, R. (1992). *Microeconomics.* Second edition, Longman, London.
5. Kanger, S. (1957) *New Foundations for Ethical Theory.* Part 1. Stockholm, 1957 (Reprinted in *Deontic Logic: Introductory and Systematic Readings,* ed. R. Hilpinen, Dordrecht, 1971, pp. 36-58.)
6. Kanger, S., Kanger, H. (1966). "Rights and Parliamentarism ", *Theoria* 32: 85-115.
7. Larsson P, Lehnbom P, Ahonen-Jonnarth U, Odelstad J. (2004). Simlation of young forest stands. Description of the program Forest stands simulation version 1.0. (In Swedish) *Reports from Creativ Media Lab,* University of Gävle, Report 2004:2, 11 pp.
8. Lindahl, L. (1977). *Position and Change. A Study in Law and Logic.* Dordrecht: Reidel.
9. Lindahl, L. & Odelstad, J. (2003). Normative Systems and Their Revision: An Algebraic Approach. *Artificial Intelligence and Law*, 11, 81-104.
10. Lindahl, L. & Odelstad, J. (2004). Normative Positions within an Algebraic Approach to Normative Systems. *Journal Of Applied Logic* 2, 63-91.
11. Lindahl, L. and Odelstad, J. (2006a). Intermediate Concepts in Normative Systems. In L. Goble and J-J. Ch. Meyer (eds.) *Deontic Logic and Artificial Normative Systems.* (DEON 2006). Berlin: Springer.
12. Lindahl, L. and Odelstad, J. (2006b). Open and Closed Intermediaries in Normative Systems. In T.M. van Engers (ed.) *Legal Knowledge and Information Systems.* (Jurix 2006). Amsterdam: IOS Press.
13. Makinson, D. (1986). On the Formal Representation of Right Relations. *Journal of Philosophical Logic* 15:403-425.
14. Odelstad, J. & Boman, M. (2004). Algebras for Agent Norm-Regulation. *Annals of Mathematics and Artificial Intelligence,* 42: 141-166, 2004.
15. Odelstad, J. & Lindahl, L. (2002). The Role of Connections as Minimal Norms in Normative Systems. *Legal Knowledge and Information Systems.* Eds. T. Bench-Capon, A. Daskalopulu and R. Winkels. Amsterdam: IOS Press.
16. Russell, S. & Norvig, P (2003). *Artificial Intelligence. A Modern Approach.* Second edition. Prentice Hall, 2003.
17. Vestlund K, Nordfjell T, Eliasson L and Karlsson A. (2005). A decision support system for selective cleaning. In: Vestlund K. 2005. *Aspects of automation of selective cleaning.* Acta Universitatits Agriculturae Sueciae 2005:74. Department of Silviculture, Swedish University of Agricultural Sciences, Umeå, Sweden. 54. p. ISBN 91-576-6973-2.