In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, Nagoya, Japan, 600-605, 1997.

Cooperation Structures

Mark d'Inverno
School of Computer Science
University of Westminster
London W1M 8JS, UK
dinverm@wmin.ac.uk

Michael Luck
Dept. of Computer Science
University of Warwick
Coventry CV4 7AL, UK
mikeluck@dcs.warwick.ac.uk

Michael Wooldridge
Agent Systems Group
Zuno Ltd, International House
London W5 5DB, UK
mjw@dlib.com

Abstract

In order to cooperate effectively with its peers, an agent must manipulate representations of the social structures in which it plays a part. The purpose of this paper is to investigate the mathematical and computational aspects of this social reasoning process. We begin by defining an abstract representation of *cooperation structures*, wherein agents cooperate to achieve goals on each other's behalf. We then investigate the question of whether or not cooperation is feasible with respect to an agent's goal, and we show that answering this question is an NP-complete problem. Finally, we investigate the conditions under which such structures can be composed to form larger structures.

1 Introduction

Cooperation is perhaps the paradigm example of social activity in both real and artificial social systems; it is certainly the best studied process in multi-agent systems research. Cooperation in human societies is an intricate and subtle activity, which has defied many attempts to formalise it. However, some progress has been made on understanding the types of situation in which cooperation can arise, and how it can proceed.

Central to the study of cooperation is the notion of a *social structure*. A social structure is a set of relations that hold between agents in a society. These relations define the dependencies that exist between agents (e.g., [Castelfranchi, 1990; d'Inverno and Luck, 1996a; 1996b]), and determine the rights and responsibilities of each agent in the society with respect to its peers. In order to cooperate effectively with its peers, an agent must *represent* any social structures in which it plays a part, and *reason* with these representations. This reasoning process is carried out in order to answer such questions as whether cooperation is possible, and to investigate how an agent stands in relation to other agents in the society. In multi-agent systems, the representation of social structures is a central research issue. For example, Durfee has developed representations of multi-agent activity known as *partial global plans* [Durfee and Lesser, 1987]. These structures can be manipulated by an agent in order to find more efficient routes to solving complex multi-agent problems.

Much work on representing social structures in multi-agent systems has been purely formal, with no obvious, direct route to implementation. Game-theoretic and economic-theoretic studies of social behaviour fall into this category [Rosenschein and Genesereth, 1985]. Although such work is central to our understanding of cooperation, it has little to tell us about the *computational* aspects of social reasoning, such as what types of social reasoning are tractable. Our aims in this paper are therefore threefold:

- to introduce an abstract symbolic representation of social structures;
- to identify and formally define some basic reasoning problems associated with these social structures; and finally
- to investigate the computational complexity of these problems.

We begin in the following section by introducing a simple, general formal framework, which can be used to represent a wide range of multi-agent scenarios. This framework defines agents as systems that have the ability to achieve certain goals, and that are capable of making independent decisions about how they will interact with other agents. We informally discuss the properties of cooperation and cooperation structures, and formally define cooperation structures within our framework. We then introduce COOPSAT, a key decision problem in social reasoning. COOPSAT is the problem of determining, given some society of agents and a particular agent's goal, whether or not cooperation is *in principle* possible to achieve the agent's goal. We show that the problem is NP-complete, and that it cannot therefore be answered in practice. We then address the issue of *manipulating* social structures, and the conditions under which they can be combined. Finally, we discuss related work, and present some conclusions and future research directions.

2 Cooperation Structures

Before we can define cooperation and cooperation structures, we need a formal framework within which we can express the definitions. A number of such formal frameworks have previously been developed, the most obvious of which being game-theory (e.g., [Rosenschein and Genesereth, 1985]) and multi-modal logic (e.g., [Wooldridge and Jennings, 1994]).

With respect to the former, the models derived are by nature quantitative rather than symbolic, and hence not well-suited to representation within a computer system. With respect to the latter, the models derived tend to be rather arcane, and do not easily lend themselves to, for example, complexity-theoretic analysis. For these reasons, we develop a simple formal framework for expressing multi-agent scenarios, using a notation based on the Z specification language [Spivey, 1992] which, in turn, is based on set theory and first-order logic. In Z, a relation R is defined as a set of ordered pairs. The expression dom R represents the set of the first elements of each of the ordered pairs of R, and ran R represents the set of second elements. Also, R^+ is the transitive closure, and R^* is the reflexive transitive closure of R.

We start by assuming a fixed, finite set Ag of agents. We use i, j, k as variables ranging over Ag. The main assumptions that we make with respect to agents are that they are *autonomous* (in that they are not benevolent), and that they have *capabilities* (in that they have the ability to achieve goals). The properties of agents are discussed in more detail elsewhere [Wooldridge and Jennings, 1995; Luck and d'Inverno, 1995].

Next, we assume a fixed, finite set G of *goals*. We use g with annotations (g', $g_1, \ldots$) as variables ranging over G. Whereas the assumption that Ag is finite seems intuitively reasonable, it may seem odd to assume that the set of goals is finite: it is common in AI to represent goals as logical formulae, and hence to allow the set of goals to be infinite in size. We make this assumption in the interests of simplicity. In this paper, we are not concerned with the question of what a goal *is* — the contents of G are left undefined (but have been considered elsewhere [Luck and d'Inverno, 1995]). However, it seems essential to introduce some notion of *consistency* between goals. We thus assume a relation $con \subseteq G \times G$, such that if $(g, g') \in con$, then g and g' are said to be consistent with one another. We write $con(g, g')$ to indicate that $(g, g') \in con$. The intuitive meaning of $con(g, g')$ is that g being satisfied does not preclude g' being satisfied: the two goals are not mutually exclusive. We shall not give a formal semantics to con, but we do require that this relation satisfies the following properties:

- reflexive: $\forall g \in G \bullet con(g, g)$; and
- symmetric: $\forall g, g' \in G \bullet con(g, g') \Leftrightarrow con(g', g)$.

In addition to consistency, we assume that goals are related to each other through a *sub-goal* relation: $\leq \; \subseteq G \times G$. A sub-goal is a component goal of another, higher-order, goal. Thus if $(g, g') \in \; \leq$ (written $g \leq g'$), then g is said to be a *sub-goal* of g'.

The $\leq$ relation must satisfy the following properties:

- reflexive:
$$\forall g \in G \bullet g \leq g;$$
- transitive:
$$\forall g, g', g'' \in G \bullet g \leq g' \wedge g' \leq g'' \Rightarrow g \leq g'';$$
- well-founded:
$$\forall g \in G \bullet \#\{g' \mid g' \leq g\} \in \mathbb{N}.$$

The first two conditions are intuitively obvious; well-foundedness simply states that no goal has an infinite, non-terminating chain of sub-goals.

We define the *strict sub-goal* relation, $<$, in the obvious way: $g < g' \Leftrightarrow g \leq g' \wedge g \neq g'$. With the exception of reflexivity, $<$ enjoys all the properties of $\leq$ and, in addition, $<$ is asymmetric. Since $<$ is well-founded, some goals are primitive and have no strict sub-goals. We write $\perp < g$ to indicate that g has no strict sub-goals. We assume that there is just one way to *achieve* a goal, by achieving all its strict sub-goals. We make this assumption in the interests of simplicity though, of course, the real-world is somewhat more complicated than this. If I have a goal of drinking tea, I have many different possible courses of action available to me: for example, I can make the tea myself, ask someone to make me a cup, or go to a cafe.

Another useful extension to $\leq$ is the *immediate sub-goal* relation: $\lll$. This relation is defined as follows: $g \lll g' \Leftrightarrow g < g' \wedge (\neg \exists g'' \in G \bullet ((g < g'') \wedge (g'' < g')))$. The function $isg : G \rightarrow \mathbb{P}\, G$ takes a goal and returns the set of all its *immediate* sub-goals: $isg(g) = \{g' \mid g' \lll g\}$.

We noted above that benevolence is not assumed in our framework. Crudely, the benevolence assumption states that agents will always attempt to do what is requested of them: they are not autonomous [Rosenschein and Genesereth, 1985]. While benevolence is reasonable for many distributed problem-solving systems, it is not an appropriate assumption in most multi-agent scenarios. In order to capture autonomy in our framework, we make use of a *will-adopt* function, $will : Ag \times Ag \rightarrow \mathbb{P}\, G$. The idea is that agent i will adopt a goal g on behalf of another agent j *iff* $g \in will(i, j)$. Where there can be no confusion, we write $will(i, g, j)$ to indicate $g \in will(i, j)$. We do not give a formal semantics to this relation, as its properties will be domain specific[1].

The *capabilities* of agents are represented in a function $cap : Ag \rightarrow \mathbb{P}\, G$. The idea is that $cap(i)$ represents the set of goals that agent i can achieve in isolation. We require the following invariant to hold between the cap and $will$ functions: $will(i, g, j) \Rightarrow g \in cap(i)$.

The various sets and relations introduced above together comprise a *framework*.

Definition 1 *A* framework *is a 6-tuple*

$$\langle Ag, G, con, \leq, will, cap \rangle$$

with components as above. Let Fr be the set of all frameworks. We use F with annotations ($F', F_1, \ldots$) as variables ranging over Fr.

For most of this paper, the framework is assumed to be fixed and understood.

2.1 Defining Cooperation Structures

Previously, Luck and d'Inverno [1996] have taxonomised the types of interactions that occur between agents in a multi-

[1] There are certain properties that it seems reasonable to demand of $will$. For example, we might specify that if i will adopt g for j, then i will also adopt any sub-goal of g for j: $\forall i, j \in Ag \bullet \forall g, g' \in G \bullet (g \in will(i, j) \wedge g' \leq g) \Rightarrow g' \in will(i, j)$. However, none of these properties are essential for our framework.

agent system, distinguishing in particular between *engagements* of non-autonomous agents and *cooperation* between autonomous (motivated) agents. In this view, autonomous agents will only adopt goals if it is to their motivational advantage to do so, while non-autonomous agents may benevolently adopt goals. Though we focus on autonomous agents, the discussion of cooperation structures below does not refer to the *reasons* for goal adoption, nor to the autonomy or non-autonomy of the agents involved and, consequently, these structures are generic and may be applied to cooperations and engagements, as described here, equally.

We define cooperation by one agent with another to mean that the agent will adopt a goal on behalf of that other agent. These cooperations give rise to a graph structure, with nodes in the graph corresponding to agents, and arcs in the graph corresponding to cooperations, labelled with goals. This leads us to the definition of a *structure*.

Definition 2 *A* structure *is a pair* (C, l) *where:*

- $C \subseteq Ag \times Ag$ *is a binary* cooperates *relation;*
- $l : C \rightarrow G$ *labels each arc in C with a goal.*

If (C, l) is a structure, then we write $C(i, j)$ to indicate $(i, j) \in C$. The intuitive interpretation of $C(i, j)$ is that agent i has delegated goal $l(i, j)$ to agent j, and thus that j is cooperating with i over this goal.

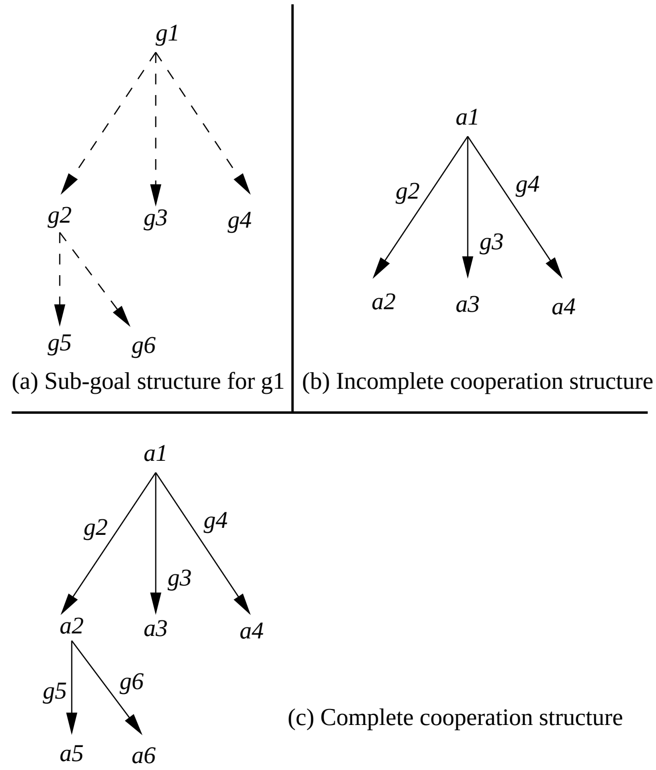
Not all structures are cooperation structures, however. For a structure to be a cooperation structure, there must first be at least two agents cooperating, and each agent in the structure must be connected to another through a cooperation over goals between them. Furthermore, agents cannot delegate goals to others who, in turn, delegate them back to the original agent. Hence there are no cycles, and no agent cooperates with itself. Also, if one agent j cooperates with another agent i over some goal g, and another agent k cooperates with j for goal g' then g' must be a sub-goal of g. Thus when agents delegate, they delegate sub-goals. Finally, all goals in a cooperation structure must be consistent with each other. These considerations lead to the following definition.

Definition 3 *A structure* (C, l) *is a* cooperation structure *iff:*

1. *C is non-empty:* $C \neq \emptyset$*;*
2. *C is weakly connected;*
3. *C is acyclic:* $\forall\, i \in Ag \bullet \neg\, C^{+}(i, i)$*;*
4. *C is irreflexive:* $\forall\, i \in Ag \bullet \neg\, C(i, i)$*;*
5. *agents delegate sub-goals:* $\forall\, i, j, k \in Ag \bullet C(i, j) \wedge C(j, k) \Rightarrow l(j, k) \leq l(i, j)$*;*
6. *goals within a structure are mutually consistent:* $\forall\, g, g' \in \operatorname{ran} l \bullet con(g, g')$*; and*
7. *j cooperates with i only if j is willing to do so:* $C(i, j) \Rightarrow will(j, l(i, j), i)$*.*

Let Coop be the set of all cooperation structures.

Condition (3) may seem too strong. To see why, consider the following scenario: John asks Paul to make some tea. Paul agrees, but asks that John boil the kettle. This kind of scenario (where i delegates a goal g to j, and j in return delegates a strict sub-goal of g back to i) occurs frequently in



(a) Sub-goal structure for g1 (b) Incomplete cooperation structure

(c) Complete cooperation structure

Figure 1: Completeness

real life. However, the problem of determining whether an arbitrary, possibly cyclic structure was in fact a legal cooperation structure would then become much harder. In addition, determining whether it was possible to fuse two cooperation structures (a problem we consider below) would also be much more complicated. For these reasons, cooperation structures are required to be acyclic.

Finally, the function, $deleg : Ag \times Coop \rightarrow \mathbb{P}\, G$, returns the set of goals that an agent delegates in some cooperation structure: $deleg(i, (C, l)) = \{l(i, j) \mid C(i, j)\}$.

2.2 Complete Cooperation Structures

Consider the following scenario. Agent a_1 wants to achieve goal g_1, but $g_1 \notin cap(a_1)$. The complete sub-goal structure for g_1 is illustrated in Figure 1(a). Agent a_1 delegates goals g_2, g_3, and g_4 to agents a_2, a_3, and a_4 respectively. Agents a_3 and a_4 have the capabilities to achieve their respective goals (i.e., $g_3 \in cap(a_3)$ and $g_4 \in cap(a_4)$), but a_2 is not capable of g_2. It should therefore delegate the sub-goals g_5 and g_6 to other agents, but it does not. So g_2 will not be achieved, and hence neither will g_1. The cooperation structure in Figure 1(b) is thus in some sense *incomplete*.

This leads to the idea of a cooperation structure being *complete* for some agent-goal pair. Informally, a cooperation structure is said to be complete for agent i and goal g *iff* either:

1. agent i has been delegated the goal g, and i is capable of g; or else

2. agent i has delegated each immediate sub-goal g' of g to some agent j, and (C, l) is complete for agent j and goal g'.

Completeness is hence a recursive notion, with the first clause as the base. It is not difficult to see that the cooperation structure in Figure 1(b) is incomplete according to this definition, but that Figure 1(c) *is* complete (assuming that $g_5 \in cap(a_5)$ and $g_6 \in cap(a_6)$). Formally, completeness is defined as follows.

Definition 4 *A cooperation structure $c = (C, l)$ is said to be* complete *with respect to agent i and goal g iff either:*

1. *$g \in cap(i)$, and for some $j \in Ag$, we have $C(j, i)$ and $l(j, i) = g$; or else*
2. *$isg(g) \subseteq deleg(i, (C, l))$, and $\forall j \in Ag$, if $C(i, j)$ and $l(i, j) \ll g$, then (C, l) is complete for agent j and goal $l(i, j)$.*

In a complete cooperation structure, there are no sub-goals left dangling: all sub-goals are successfully delegated and hence, by the intuitive semantics for the sub-goal relation, every goal in the structure is achieved.

3 Is Cooperation Possible?

Suppose an agent has a goal that it wants to achieve, and further suppose that the agent either cannot achieve the goal in isolation (because it does not have the resources), or does not want to achieve it in isolation (because in so doing, it would clobber one of its other goals) [Wooldridge and Jennings, 1994]. The obvious question this agent should ask is: can I get other agents to help me with this goal? This is a *satisfiability* problem, similar in nature to the question of whether a formula of some particular logic is true under some interpretation. Formally, the problem can be stated as follows.

Definition 5 *(The* COOPSAT *problem.) Given a framework $F = \langle Ag, G, con, \leq, will, cap\rangle$, an agent $i \in Ag$, and a goal $g \in G$, does there exist a cooperation structure over F that is complete for (i, g)?*

Theorem 1 COOPSAT *is* NP*-complete.*

Proof: Membership of NP is easy: given an instance $\langle\langle Ag, G, con, \leq, will, cap\rangle, i, g\rangle$ of the COOPSAT problem, simply guess a cooperation structure (C, l) that is complete for (i, g). The size of the structure is bounded above by $\#(Ag \times Ag)$ and, since Ag is finite, guessing can be done in polynomial time. Verifying that (C, l) is complete for (i, g) can also be done in polynomial time.

For completeness, we must show that COOPSAT is in some sense no easier than all other NP-complete problems. To do this, it suffices to show that any instance I of some known NP-complete problem can be transformed into an instance of $\tau(I)$ of COOPSAT such that the transformation can be done in polynomial time, and the transformed problem $\tau(I)$ has a solution only if the original problem I has a solution. For COOPSAT, we define a reduction from a version of the well-known HAMILTONIAN CYCLE (HC) problem.

An instance of HC is determined by a graph $(N, A \subseteq N \times N)$. The aim is to answer 'yes' if A has a cycle containing all nodes without repetition, 'no' otherwise. The idea behind the reduction is to encode the relation A in the *will* relation, and the requirement for the cycle in the sub-goal relation $<$.

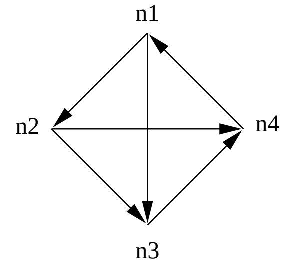


Figure 2: Illustrating the Reduction

To see how the reduction works, consider the following graph G_1 (illustrated in Figure 2):

$$G_1 = (\{n_1, n_2, n_3, n_4\}, \{(n_1, n_2), (n_2, n_3), (n_3, n_4), (n_4, n_1), (n_1, n_3), (n_2, n_4)\})$$

This graph has a Hamiltonian cycle (n_1, n_2, n_3, n_4). We transform G_1 into an instance $\tau(G_1)$ of COOPSAT. To do this, we first create five goals $g_0, \ldots, g_4$, and five agents, $n_1, \ldots, n_4, end$. We then create a linear sub-goal structure for g_0:

$$g_4 \ll g_3 \ll g_2 \ll g_1 \ll g_0$$

The *will* relation is then generated as follows.

From \ To	n_1	n_2	n_3	n_4	end
n_1					
n_2	$g_1, \ldots, g_4$				
n_3	$g_1, \ldots, g_4$	$g_1, \ldots, g_4$			
n_4		$g_1, \ldots, g_4$	$g_1, \ldots, g_4$		
end				$g_1, \ldots, g_4$	

Now consider the COOPSAT problem determined by this framework and the agent-goal pair (n_1, g_0). The problem clearly has a solution, which will look like that below, where each question mark represents an agent.

$$n_1 \xrightarrow{g_1} ? \xrightarrow{g_2} ? \xrightarrow{g_3} ? \xrightarrow{g_4} ?$$

In fact, this problem has a simple solution:

$$n_1 \xrightarrow{g_1} n_2 \xrightarrow{g_2} n_3 \xrightarrow{g_3} n_4 \xrightarrow{g_4} end$$

The transformation from HC to COOPSAT is entirely automatic, (see Figure 3) and is polynomial. We leave it for the reader to see that the generated COOPSAT problem has a solution only if the original HC problem does, and so we are done. ◻

Suppose we had an algorithm that was guaranteed to give us the correct answer to an instance of the COOPSAT problem. This algorithm would take as input a framework, an agent and a goal and, some time later, would be guaranteed to generate as output of either 'yes' (indicating that a solution did indeed exist), or 'no' (indicating that the problem had no solution). Now, suppose the algorithm answered 'yes'. Then the agent would know that cooperation was possible. It would then have to start delegating goals to other agents. But the simple

```
let n = #N be the size of N
for each node i ∈ N, create a corresponding agent i
create a 'dummy' agent, called end
create n + 1 goals, g_0, ..., g_n
set con(g, g') for all goals g, g'
define < by g_n ≪ ··· ≪ g_1 ≪ g_0
for each agent j ≠ end
    for each agent k ≠ end
        if (j, k) ∈ A then
            if k = n_1 then
                will := will ∪ {(j, end) ↦ {g_n}}
            else
                will := will ∪ {(j, k) ↦ {g_1, ..., g_n}}
        else
            will := will ∪ {(j, k) ↦ ∅}
    end-for
end-for
cap := {end ↦ {g_n}}
```

Figure 3: Reducing HC to COOPSAT

'yes' answer does not indicate exactly who the agent should delegate *to*. A more useful algorithm would not only say 'yes', but would also *produce* a solution to the problem. This leads us to the following, closely related problem.

Definition 6 *(The* COOPFIND *problem.) Given a framework* $F = \langle Ag, G, con, \leq, will, cap\rangle$*, an agent* $i \in Ag$*, and a goal* $g \in G$*, find a cooperation structure over F that is complete for* (i, g) *if such a structure exists, or else answer that there is no solution.*

This problem is clearly related to many similar planning problems [Allen *et al.*, 1990]; it is also naturally viewed as a type of constraint satisfaction problem.

4 Composing Cooperation Structures

In a real society there will be many cooperation structures in existence at any given time. This may result in redundancy through different agents achieving the same goals in different contexts. In order to remove this redundancy it may be possible to *compose* two cooperation structures. Given two cooperation structures, $c = (C, l)$ and $c' = (C', l')$, we write $c \cup c'$ to denote the set-theoretic union $(C \cup C', l \cup l')$ of these structures. However, this composition can only occur in certain situations: we cannot expect the union of two arbitrary cooperation structures to be a legal structure. For example, if we have some arc, (i, j), that appears in both C and C', but $l(i, j) \neq l'(i, j)$, then $l \cup l'$ is not a function (since $(l \cup l')(i, j)$ is not well-defined). This raises the question of what conditions are required for the union of two cooperation structures itself to be a cooperation structure. It turns out that we can define these conditions precisely. First, we extend the notion of *intra*-structure goal consistency to *inter*-structure goal consistency.

Definition 7 *Let* $c = (C, l)$ *and* $c' = (C', l')$ *be cooperation structures. Then c and c′ are said to be* consistent *(written* $cons(c, c')$*) iff* $\forall g \in \operatorname{ran} l \bullet \forall g' \in \operatorname{ran} l' \bullet con(g, g')$.

We can now define what it means for two structures to be *compatible*.

Definition 8 *If* $c = (C, l)$ *and* $c' = (C', l')$ *are cooperation structures, then c and c′ are said to be* compatible *(written* $compat(c, c')$*) iff:*

1. $C \cup C'$ *is weakly connected;*
2. $C \cup C'$ *is acyclic;*
3. *the two structures agree on labels:* $\forall i, j \in Ag \bullet C(i, j) \wedge C'(i, j) \Rightarrow l(i, j) = l'(i, j)$
4. $\forall i, j, k \in Ag$,
   (a) *if* $C(i, j) \wedge \neg C(j, k) \wedge \neg C'(i, j) \wedge C'(j, k)$ *then* $l'(j, k) \leq l(i, j)$;
   (b) *if* $C'(i, j) \wedge \neg C'(j, k) \wedge \neg C(i, j) \wedge C(j, k)$ *then* $l(j, k) \leq l'(i, j)$*; and*
5. $cons(c, c')$.

One might expect that compatibility is an equivalence relation over the set of all cooperation structures, but this is not in fact the case, as the following theorem establishes.

Theorem 2 *The compatibility relation is reflexive and symmetric, but not transitive.*

Proof: Reflexivity and symmetry are obvious. For transitivity, it is easy to construct a counter-example. Suppose we had three cooperation structures c, c', c''. Further suppose that $compat(c, c')$ and $compat(c', c'')$, and that c and c' share a single agent i in common, and c' and c'' share a different single agent j in common ($i \neq j$). Hence c and c'' are disjoint, so $c \cup c''$ is not weakly connected. Hence c and c'' are not compatible. □

Determining whether two cooperation structures are compatible is a tractable problem.

Theorem 3 *It is possible to determine whether two cooperation structures are compatible in polynomial time — no worse than* $O(\#(Ag)^3)$.

Proof: The only non-trivial step involves showing that the resulting cooperation relation is acyclic, which requires checking the transitive closure of the relation — using Warshall's algorithm, the transitive closure can be computed in time $O(\#(Ag)^3)$ [van Leeuwen, 1990, pp540–544]. The overall time complexity is therefore no worse than $O(\#(Ag)^3)$. □

Theorem 4 *If* $c = (C, l)$ *and* $c' = (C', l')$ *are cooperation structures, then* $c \cup c'$ *is a cooperation structure iff* $compat(c, c')$.

Proof: (Omitted due to lack of space.) □

Once we know that two cooperation structures are compatible, the problem of generating their union is computationally trivial.

5 Related Work

As we noted in Section 1, cooperation is a widely studied issue in multi-agent systems research. Despite this, we are aware of little other work that considers cooperation in the complexity-theoretic way proposed in this paper. Probably the most closely related work to ours is that of Shehory and Kraus on *coalition formation* [Shehory and Kraus, 1996]. Coalition formation is the process of devising a team of agents to work on a goal, and is rather similar to our COOPSAT problem. The most obvious differences between our work and that of Shehory and Kraus are that they assume benevolent agents, and they present algorithms (adapted from the *set covering problem*) to design coalitions. In addition, the work of Tennenholtz and Moses on the *multi-entity* model of multi-agent systems [Tennenholtz and Moses, 1989] is also closely related. This model is used to define the *cooperative goal achievement* (CGA) problem, which can crudely be stated as: given a set of benevolent agents, each with their own goals, is there some plan for the set that will achieve all their goals? Tennenholtz and Moses show that this problem is PSPACE-complete. Our framework most significantly differs from theirs in that they allow a richer representation of goals (as arbitrary propositional logic formulae) and, in addition, they also assume benevolence.

6 Conclusion

Cooperation is a key process for multi-agent systems research and, as such, it has received a considerable amount of attention in the multi-agent systems literature. However, mathematical treatments of cooperation have focussed primarily on either game-theoretic or modal logic formulations.

When many agents cooperate together, a cooperation structure emerges, which can be represented formally as a directed graph with certain properties. In this paper, we have formally defined the properties that must hold of such a graph to be considered as a cooperation structure. We have shown that, even when making simplifying and limiting assumptions about the world, the problem of determining whether cooperation structures are available to achieve an agent's goal is NP-complete. The problem of computational complexity and tractability has often been overlooked in the design of multi-agent systems. In future work we wish to use our framework in order to formally define other key social reasoning problems, and analyse the computational complexity of such problems. In addition, we aim to investigate algorithms for solving these problems.

References

[Allen *et al.*, 1990] J. F. Allen, J. Hendler, and A. Tate, editors. *Readings in Planning*. Morgan Kaufmann, San Mateo, CA, 1990.

[Castelfranchi, 1990] C. Castelfranchi. Social power. In Y. Demazeau and J.-P. Müller, editors, *Decentralized AI — Proceedings of the First European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-89)*, pages 49–62. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands, 1990.

[d'Inverno and Luck, 1996a] M. d'Inverno and M. Luck. A formal view of social dependence networks. In *Distributed Artificial Intelligence: Architecture and Modelling, Lecture Notes in Artificial Intelligence, 1087*, pages 115–129. Springer Verlag, 1996.

[d'Inverno and Luck, 1996b] M. d'Inverno and M. Luck. Formalising the contract net as a goal directed system. In W. Van de Velde and J. W. Perram, editors, *Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-96), Lecture Notes in Artificial Intelligence Volume 1038*, pages 72–85. Springer-Verlag, 1996.

[Durfee and Lesser, 1987] E. H. Durfee and V. R. Lesser. Using partial global plans to coordinate distributed problem solvers. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87)*, pages 875–883, 1987.

[Luck and d'Inverno, 1995] M. Luck and M. d'Inverno. A formal framework for agency and autonomy. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 254–260. AAAI Press / MIT Press, 1995.

[Luck and d'Inverno, 1996] M. Luck and M. d'Inverno. Engagement and cooperation in motivated agent modelling. In *Distributed Artificial Intelligence: Architecture and Modelling, Lecture Notes in Artificial Intelligence, 1087*, pages 70–84. Springer-Verlag, 1996.

[Rosenschein and Genesereth, 1985] J. S. Rosenschein and M. R. Genesereth. Deals among rational agents. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI-85)*, pages 91–99, 1985.

[Shehory and Kraus, 1996] O. Shehory and S. Kraus. Formation of overlapping coalitions for precedence-ordered task-execution among autonomous agents. In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96)*, pages 330–337. AAAI Press / MIT Press, 1996.

[Spivey, 1992] J. M. Spivey. *The Z Notation*. Prentice Hall, Hemel Hempstead, 2nd edition, 1992.

[Tennenholtz and Moses, 1989] M. Tennenholtz and Y. Moses. On cooperation in a multi-entity model: Preliminary report. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 918–923, 1989.

[van Leeuwen, 1990] J. van Leeuwen. Graph algorithms. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 525–631. Elsevier Science Publishers B.V., 1990.

[Wooldridge and Jennings, 1994] M. Wooldridge and N. R. Jennings. Formalizing the cooperative problem solving process. In *Proceedings of the Thirteenth International Workshop on Distributed Artificial Intelligence (IWDAI-94)*, pages 403–417, July 1994.

[Wooldridge and Jennings, 1995] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.