Towards a Sheaf Semantics for Cooperating Agents Scenarios

Viorica Sofronie*

RISC-Linz, Johannes Kepler University, A-4040 Linz, Austria
e-mail Viorica.Sofronie@risc.uni-linz.ac.at

Abstract. The ultimate goal of our work is to show how sheaf theory can be used for studying cooperating robotics scenarios. In this paper we propose a formal definition for systems and define a category of systems. The main idea of the paper is that relationships between systems can be expressed by a suitable Grothendieck topology on the category of systems. We show that states and (parallel) actions can be expressed by sheaves and use this in order to study the behavior of systems in time.

1 Introduction

Originally, the theory of sheaves was conceived as a tool in topology and algebraic geometry. The influence of sheaf theory has spread in many areas of mathematics: besides the fields where its origins are, such as analysis, topology and algebraic geometry, it is now widely used in algebra (representations of algebras by continuous sections; global subdirect products, see e.g. [KC79]), as well as logic (cf. [FS79] and related work – for details see also [MLM92]). Sheaf theory was developed in mathematics because of the necessity of studying the relationship between "local" and "global" phenomena.

In what follows we will briefly explain why we think that sheaf theory can be a useful framework for modeling cooperating agents scenarios.

When modeling states, actions or behavior it is often necessary to make a link between "local" properties (characteristic for given subsystems) and "global" properties (relevant for the whole system). The goal of our study is an analysis of subsystem interaction, taking into account the contribution of subsystems to the description (states, actions, behavior) of the whole system. The alternance "local - global" that occurs in this case suggests that a sheaf-theoretical approach to the study of systems of cooperating agents (and in the study of concurrency in general) would be natural. Moreover, the tools of sheaf theory, and of topos theory in general, in particular geometric logic, should explain why some properties of systems are preserved when restricting to subsystems, and why there are cases when properties of subsystems are not transferred to the system obtained by their interconnection. This is part of ongoing work (cf. [Sof96]).

There already exist a number of approaches to concurrency based on "fiberings" and sheaf theory. Among them, [MP86], [Pfa91], [Gog92], [Lil93], [Mal94].

* Partially supported by the Austrian Science Foundation under ESPRIT BRP 6471 "MEDLAR II"

In [MP86] the authors aim at developing a structural theory of concurrency in which the locality of interaction between subsystems is described with the mathematical tools of the theory of sheaves. They show that the behavior of a given family of interconnected systems can be modeled by so-called behavior monoids (which turn to form sheaves of monoids). In [Gog92], a sheaf semantics aiming at modeling the behavior of concurrent interacting objects is presented. The approach is based on an earlier paper [Gog75]. These ideas have been applied to Petri Nets by Lilius [Lil93]. The ideas from [Gog92] have been further developed by Malcolm in [Mal94], where a formalization of object classes and systems of objects is given, in order to study basic properties of ways in which systems of objects may be interconnected. He expresses the hope that by using a more general notion of sheaf as a functor on a category with a Grothendieck topology, an adjunction between system specifications and sheaves of objects can be obtained.

In a series of papers, J. Pfalzgraf develops the idea of “logical fiberings”, with the goal of developing a (non-classical) “fibered logical calculus”, by means of which one could construct logical controllers for multi-tasking scenarios in a formal way (for details see e.g. [Pfa91], [Pfa95]). In [Pfa95], he points out that the notion of a fibering is closely related to indexed systems (indexed categories) and to sheaves (cf. their relation with topos theory and the geometric aspects). These methods and concepts arose in the process of modeling cooperating agents scenarios (see [Pfa91]) and have been illustrated, in the frame of the MEDLAR project, on concrete examples (see for example [Pfa93],[PS95], [PSS95], [PSS96]).

Finally, we would like to point out another application of sheaf theory in computer science in a paper by Srinivas (cf. [Sri93]). Although the topic of [Sri93] (pattern matching) has no link with our approach, the idea of using Grothendieck topologies occurred to us after reading that paper.

In this paper we make first steps towards a theoretical approach, based on sheaf theory, for describing systems of cooperating agents. The basic idea of our formalism is that even relatively simple agents, such as a robot that provides an assembly bench with pieces, are in fact complex systems composed of interacting subsystems, like joints and wrists, locomotion modules, etc. Therefore, instead of “individual” agents, a category of “systems” is considered, where the morphisms describe the relation “is a subsystem of”. We show that under certain (non-restrictive) hypotheses a Grothendieck topology (describing a “covering” relation between systems) can be defined on this category, and that states as well as admissible parallel actions can be modeled by sheaves. We use these results for studying the behavior in time of the systems, adapting one of the ideas presented in [Gog92]. The difference between our approach and the approach of Monteiro and Pereira in [MP86] lies in the fact that here we do not model the behavior of interconnected systems by sheaves of monoids. These aspects are analyzed in [Sof96].

The paper is structured as follows:

In section 2, as a motivation for our theoretical study, we illustrate the problems that appear by means of a simple example, adapted from [Pfa93]. This example leads to a formal definition of a system, given in section 3.

In section 3 we give a formal definition of a system and introduce a category of systems on which a Grothendieck topology is defined. We show that states and parallel actions can be defined as sheaves over this site.

In section 4 we focus on modeling the behavior of a system. We start with the formalism developed by Goguen in [Gog92], and slightly modify Goguen's definition for the behavior of a system, explicitly indicating for every moment of time not only the state of the system but also the action performed at the given moment, and obtain a contravariant functor from the category of systems to the category of sheaves over time. We also show that a contravariant functor from $\mathcal{T}$ to the category of sheaves over Sys can be defined.

The goal of the present paper is to clarify some of the basic notions required in the study of cooperating agents. At this level we do not yet aim at formalizing scheduling, especially when a synchronization of the actions is needed. In further work we will take into account the time necessary for the execution of the actions, and also precedence relations between actions; moreover we will consider space-dependency of formulas. We hope that our approach will help in clarifying some of the problems stated in [PS95] and [Pfa95].

2 An Example

We begin with a simple example (adapted from [Pfa93]) as a motivation for our theoretical study, for the definitions that will be given, and for the assumptions that will be made.

Let R_0, R_1, R_2, R_3 be four robots performing the following task:

- R_0 receives a work piece a and a work piece b and performs an assembly task. The work piece r obtained from assembling a and b is placed on the assembly bench.
- R_1 furnishes pieces of type a. He checks whether there are pieces of type a left in stock, and whether a piece of type a or an r resulting from assembling a and b is placed on the assembly bench of R_0. If there are pieces of type a in stock, and if no a or r are placed on the table, R_1 brings a piece of type a to R_0.
- R_2 furnishes pieces of type b. He checks whether there are pieces of type b left in stock, and whether a b or an r is placed on the table. If there are pieces of type b in stock, and no b or r is on the table, R_2 brings a piece of type b to R_0.
- After R_0 has assembled a and b, R_3 receives the result r and transports it to the stock.

Let S be the system resulting from the interaction of these robots. We can assume that the system can be "described" by the interconnected subsystems S_0, S_1, S_2, and S_3, which correspond to the robots R_0, R_1, R_2 and R_3.

2.1 States

The states of the system S can be expressed using the *control variables* described in the table below. The set of control variables relevant for system S_0 is $X_0 = \{p_a, p_b, p_r\}$, the one for S_1 is $X_1 = \{s_a, p_a, p_r\}$, for S_2, $X_2 = \{s_b, p_b, p_r\}$, and for S_3, $X_3 = \{p_r\}$. We will assume that the subsystems $S_0, \ldots, S_3$ communicate via common control variables. In Figure 1 we show how the control variables are shared, and how they can be used for communication with "external" systems (e.g. Stock-a, Stock-b).

Variable	Description	System
s_a = in-stock-a	"there is at least one piece of type a in stock"	S_1
s_b = in-stock-b	"there is at least one piece of type b in stock"	S_2
p_a = on-table-a	"a piece of type a is on the assembly bench"	S_0,S_1
p_b = on-table-b	"a piece of type b is on the assembly bench"	S_0,S_2
p_r = on-table-res	"the result r is on the assembly bench"	S_0, S_1, S_2, S_3

A *state* of the system S is a possible assignment of truth values to the relevant control variables. We might additionally assume that only some of these assignments are admissible, imposing some *constraints* on the values that can be taken by the control variables. This turns out to be especially useful when the control variables are not independent.

Let us assume for example that in the given system it is not allowed to have a result piece and a piece of type a or b on the working bench at the same time, but it is allowed to have a piece of type a and one of type b. That means that p_a and p_r cannot both be true, and p_b and p_r cannot both be true. This can be expressed by a set of identities on the boolean algebra freely generated by the control variables of the system, in this case $Id = \{p_a \wedge p_r = 0, p_b \wedge p_r = 0\}$.

The agents R_0 and R_1 can "communicate" using the control variables common to these systems, namely the set $X_{01} = \{p_a, p_r\}$. Analogously, the agents R_0 and R_2 can "communicate" using the set $X_{02} = \{p_b, p_r\}$, and R_1 and R_2 using the set $X_{12} = \{p_r\}(= X_3)$. We can therefore assume that the structure of the given systems of cooperating agents as an interconnection of subsystems determines a topology on the set of control variables. Also the set of constraints (if they do not link variables in different subsystems) can be "distributed" over the subsystems in the same way.

Consider the basis $\mathcal{B} = \{X_0, X_1, X_2, X_3, X_{01}, X_{02}\}$, consisting of the sets of control variables corresponding respectively to the subsystems consisting of the robots $R_0, \ldots, R_3$, as well as to their "subsystems" by means of which the communication is done. The corresponding restrictions of the set of constraints are $\{Id_0, Id_1, Id_2, Id_3, Id_{01}, Id_{02}\}$, where $Id_0 = Id$, $Id_1 = \{p_a \wedge p_r = 0\}$, $Id_2 =$

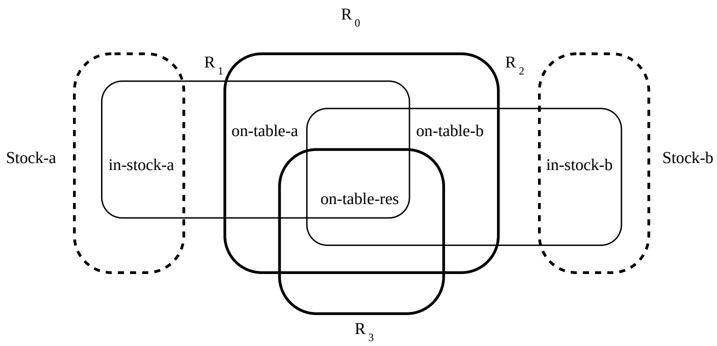

Fig. 1. Control Variables

$\{p_b \wedge p_r = 0\}$, and $Id_{01} = Id_{02} = Id_3 = \emptyset$, corresponding to the subsystems mentioned above.

The set of states of the system will be the set of those assignments of truth values to the control variables that satisfy this set of identities. Similarly, the set of states for system S_i (corresponding to the agent R_i) is $St(S_i) = \{s_i : X_i \to \{0,1\} \mid s_i \models Id_i\}$. Thus $St(S_0) = \{s_0 : X_0 \to \{0,1\} \mid s_0(p_a)s_0(p_r) = 0, s_0(p_b)s_0(p_r) = 0\}$, $St(S_1) = \{s_1 : X_1 \to \{0,1\} \mid s_1(p_a)s_1(p_r) = 0\}$, $St(S_2) = \{s_2 : X_2 \to \{0,1\} \mid s_2(p_b)s_2(p_r) = 0\}$, and $St(S_3) = \{s_3 \mid s_3 : X_3 \to \{0,1\}\}$.

It is easy to see that for every family $(s_i)_{i=0,\dots,3}$ with the property that s_i is a state for the system S_i (corresponding to R_i), and such that for every i, j, s_i and s_j coincide on the common control variables, there is exactly one state of the system, s, such that the restriction of s to the control variables P_i is s_i for every i. This means that the following gluing condition is satisfied:

- For every $\{s_B\}_{B \in \mathcal{B}}$, where $s_B : B \to \{0,1\}$ satisfies the equations in $Id_B = \{e \in Id \mid Var(e) \subseteq B\}$, such that for every $B_1, B_2 \in \mathcal{B}$, $s_{B_1|B_1 \cap B_2} = s_{B_2|B_1 \cap B_2}$, there exists a unique $s : P \to \{0,1\}$ that satisfies the set of constraints Id, such that for every $B \in \mathcal{B}$, $s_{|B} = s_B$.

Since there are typical properties of a sheaf visible, this leads to the idea that the link between local and global states could best be described by sheaves over a suitable topology on the set of control variables (or over a suitable Grothendieck topology on a category of systems) defined by the structure of the given system.

Remark: Note that the gluing property described above does not hold for every topology on the set X of control variables of the system. Consider for example the discrete topology on X. Then X can be covered by the family $\{\{s_a\}, \{s_b\}, \{p_a\}, \{p_b\}, \{p_r\}\}$. Then the following family of assignments of truth values to the control variables: $s_{\{s_a\}}(s_a) = 0$, $s_{\{s_b\}}(s_b) = 0$, $s_{\{p_a\}}(p_a) = 1$, $s_{\{p_b\}}(p_b) = 1$, $s_{\{p_r\}}(p_r) = 1$ agrees on common control variables (because the domains are disjoint), but no information about the constraints can be recovered, hence by "gluing" these mappings together one obtains a map $s : X \to \{0, 1\}$ which does not satisfy the set Id of identities. This shows that an appropriate topology on X has to respect the way the constraints are shared between subsystems.

2.2 Actions

The system S is also characterized by a set of (atomic) actions. Below we will give the list of the atomic actions (with pre- and postconditions) and the agent that performs them.

Action	Description	Precond.	Postcond.	Agent/Interpretation
A	Assemble a piece of type a with one of type b	$p_a = 1$ $p_b = 1$ $p_r = 0$	$p_a = 0$ $p_b = 0$ $p_r = 1$	R_0: assemble
B_a	Bring a piece of type a	$p_a = 0$ $s_a = 1$ $p_r = 0$	$p_a = 1$ $s_a = 0$ $p_r = 0$	R_1: give-a R_0: receive-a
B_b	Bring a piece of type b	$p_b = 0$ $s_a = 1$ $p_r = 0$	$p_b = 1$ $s_a = 0$ $p_r = 0$	R_2: give-b R_0: receive-b
T_r	Store the result	$p_r = 1$	$p_r = 0$	R_3: receive-r R_0: give-r

We can assume for example that R_1 and R_2 can perform the actions of bringing a piece of type a respectively b in parallel but that R_0 is not allowed to execute in parallel the action of taking a piece of type a and of giving the result to R_3. We also can assume that other actions, as for example give-a (by R_1) and receive-a (by R_0) have to be executed in the same time. Therefore, they have been "identified" in the larger subsystem under the name B_a. Figure 2 shows how actions are shared between subsystems, and some relations between them (in what follows for the sake of simplicity we do not consider the actions get-a, get-b and transport). It is also natural to suppose that some *constraints* are given, expressing which of these actions can be performed in parallel, which cannot, which must be executed in the same time, and so on.

There are many possibilities to specify this kind of constraints. One solution is to consider parallel actions as subsets of the set of atomic actions A, or equivalently by maps $f : A \to \{0, 1\}$, where $f(a) = 1$ means that a is executed and $f(a) = 0$ means that a is not executed. In this case the constraints can be described by imposing restrictions on the combinations of the values (0 or 1) that can be assigned to the atomic actions in an admissible parallel action. This approach is very similar to the one adopted in the description of states (section 2.1). We will assume that the constraints can be described by identities on the boolean algebra freely generated by the set of atomic actions A. (On

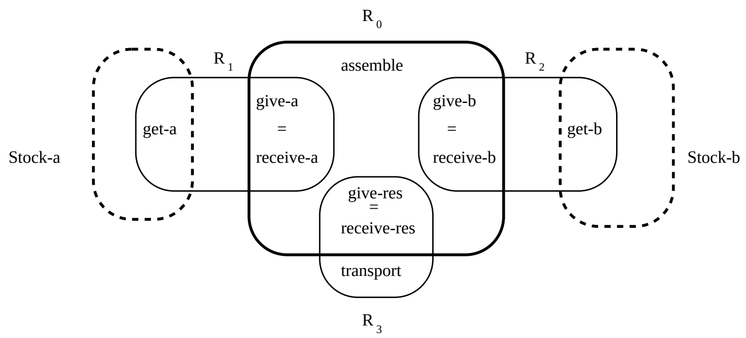

Fig. 2. Actions

our example, among the constraints are: $\mathsf{give\text{-}a} = \mathsf{receive\text{-}a}$, $\mathsf{give\text{-}b} = \mathsf{receive\text{-}b}$, $\mathsf{give\text{-}res} = \mathsf{receive\text{-}res}$, $\mathsf{give\text{-}res} \wedge \mathsf{receive\text{-}a} = 0$, $\mathsf{give\text{-}res} \wedge \mathsf{receive\text{-}b} = 0$, etc.).

There exist approaches for modeling concurrency where it is necessary to specify which actions can be performed in parallel and which not. One such approach is based on considering a "dependence" relation on the set of (atomic) actions, i.e. a reflexive and symmetric relation $D \subseteq A \times A$: the parallel execution is then only allowed for those actions which are "independent" w.r.t. D. This leads to the study of partial-commutative monoids (cf. [Die90]). The link with this approach is analyzed in [Sof96].

As when considering states, the structure of the system as an interconnection of subsystems induces a topology on the set of all actions. Correspondingly, the constraints distribute over the subsystems (it seems natural to make the assumption that all the constraints are made "inside" some specified subsystem). A basis $\mathcal{B}$ for this topology is obtained as explained in the study of states, taking the family of all actions corresponding to the subsystems $S_0, \dots, S_3$, as well as to the "subsystems" by means of which the communication is done (i.e. finite intersections of those sets). In the case where there are constraints that link actions in different subsystems (that could be for example expressed in some special "scheduling systems"), the actions that correspond to these scheduling systems will also be considered elements in the basis. Also in this case a similar gluing property holds:

– For every family of parallel actions $\{f_B\}_{B \in \mathcal{B}}$, where $f_B : A_B \to \{0,1\}$ satisfies the constraints $Id_B = \{e \in Id \mid Var(e) \subseteq B\}$, such that for every $B_1, B_2 \in \mathcal{B}$, $f_{B_1 | B_1 \cap B_2} = f_{B_2 | B_1 \cap B_2}$, there exists a unique $f : A \to \{0,1\}$ that satisfies Id such that for every $B \in \mathcal{B}$, $f_{|B} = f_B$.

Note that the fact that the gluing property holds is strongly related to the specific form of the constraints in C_A (boolean equations in our case). If for example a parallel action $f : A \to \{0,1\}$ is allowed if and only if $f^{-1}(1)$ is finite, then the infinite gluing property does not hold.

3 Systems

In what follows we will use basic notions of category theory. We will also assume known the definitions for Grothendieck topologies and sheaves with respect to a Grothendieck topology. For details see, for example, [MLM92].

Taking into account the considerations in section 2, we will assume that a system S can be described by:

- A set X of control variables of the system (for every control variable $x \in X$ a set V_x of possible values for x), and a set Γ of constraints, specifying which combinations of values for the variables are admissible (i.e. satisfy Γ). An admissible combination of values of the control variables will describe a state of the system S. The set of states of the system will be denoted $St(S)$,
- A set A of atomic actions (where for every $a \in A$, X_a denotes the minimal set of control variables a depends on and $Tr_a \subseteq St(S)_{|X_a} \times St(S)_{|X_a}$ a relation indicating how the values of these variables change when the action a is performed), together with a set C of constraints that shows which actions are incompatible and cannot be performed in parallel.

In order to formally express the "constraints" on the possible combinations of values for the control variables, we need a "language" in which these are expressed.

Definition 1. A *system* $S = (\Sigma, X, \Gamma, M, A, C)$ consists of:

(1) A language $\mathcal{L}_S$ consisting of
(1a) a signature $\Sigma = (\mathsf{Sort}, O, P)$ (Sort is the set of sorts, O the set of operation symbols, and P the set of predicate symbols),
(1b) a (many-sorted) set of variables $X = \{X_s\}_{s \in \mathsf{Sort}}$,
(2) A set $\Gamma \subseteq \mathsf{Fma}_\Sigma(X)$ (the set of constraints or axioms of S),
(3) A model M (structure of similarity type Σ),
(4) A set of actions A; for every $a \in A$ a set $X_a \subseteq X$ of variables a depends upon, and a transition relation Tr_a.
(5) A set C of constraints, expressed by boolean equations over $F_B(A)$ (the free boolean algebra generated by A) stating which actions can (or have to) be executed in parallel.

Remark: In definition 1 we fix a model M for the system S (corresponding e.g. to the real world) and we allow as constraints formulas in the many-sorted language of the system.

Given a system S, if not otherwise specified, we will refer to its signature, set of variables, constraints on the values of the variables, model, set of actions and constraints on actions by Σ_S, X_S, Γ_S, M_S, A_S, C_S respectively. If a family of systems $\{S_i\}_{i \in I}$ is given, if not otherwise specified we will refer to the signature, set of variables, constraints on the values of the variables, model, set of actions and constraints on actions of S_i by Σ_i, X_i, Γ_i, M_i, A_i, C_i, for every $i \in I$.

Systems often arise in relationship with other systems. The corresponding relationships between systems are expressed by morphisms. Obviously, there is some choice in how to define appropriate morphisms, depending on the extent of the relationship between systems we want to express.

For instance, a *category of systems* SYS can be defined, having as objects systems. A morphism f in SYS from a system S_1 to a system S_2 consist of a "translation" of the language (resp. actions) of S_1 into the the language (resp. actions) of S_2 (i.e. a family of mappings $f_\Sigma : \Sigma_1 \to \Sigma_2$, $f_X : X_1 \to X_2$, $f_A : A_1 \to A_2$) such that for every $\phi \in \Gamma_1$ (resp. in C_1) the "translation" of ϕ to the language of S_2 via f_Σ, f_X (resp. f_A) is in Γ_2 (resp. in C_2); and transitions in S_1 are mapped into transitions of S_2.

Sometimes, however, the systems are not able to communicate using such "translations": important is that the systems have common subsystems by means of which the communication is done. In what follows, we will focus on this last aspect.

Definition 2. Let S_1 and S_2 be two systems. We say that S_1 *is a subsystem of* S_2 if and only if $\Sigma_1 \subseteq \Sigma_2$, $X_1 \subseteq X_2$, $A_1 \subseteq A_2$, and the constraints in Γ_1 (resp. C_1) are consequences of the constraints in Γ_2 (resp. C_2), and the model M_1 is the restriction of the model M_2 to the signature of S_1 ($M_1 = U^{\Sigma_2}_{\Sigma_1} M_2$).

In concrete applications we usually are interested only in some subcategory Sys of SYS, having as objects those systems relevant for the given application (in the example in section 2 the relevant systems are $S_0, \dots, S_3$ together with their common subsystems, and the systems obtained by interconnecting them), and a morphism between a system S_1 and S_2 if and only if S_1 is a subsystem of S_2.

In what follows we will assume given a family InSys of interacting systems. To enforce the compatibility of models on common sorts, we may assume that all these systems are subsystems of a "universal system" S_U. We further assume that the family InSys is closed under intersections (pullbacks of subsystems of S_U) i.e. it contains all those subsystems by means of which intercommunication is done. The elements of InSys are the "building blocks" from whose interconnection larger systems arise.

Definition 3. The *category of systems*, Sys, has as objects all the systems that can be obtained by interconnecting elements in $\mathsf{In\text{-}Sys}$, and a morphism from S_1 to S_2 if and only if S_1 is a subsystem of S_2.

It is easy to see that Sys has pullbacks, namely for every $S_1, S_2 \hookrightarrow S$, $S_1 \times_S S_2 = (\Sigma_1 \cap \Sigma_2, X_1 \cap X_2, \Gamma_1 \cap \Gamma_2, M_{12}, A_1 \cap A_2, C_1 \cap C_2)$, where $M_{12} = U^{\Sigma}_{\Sigma_1 \cap \Sigma_2} M$.

Definition 4. A *covering family* for a system $S = (\Sigma, X, \Gamma, M, A, C)$ is a family $\{S_i \hookrightarrow S \mid i \in I\}$ of subsystems of S, with the property that $\bigcup_i \Sigma_i = \Sigma$, $\bigcup_i X_i = X$, $\bigcup_i A_i = A$, and such that the constraints in Γ(resp. C) are consequences of the constraints in $\bigcup_i \Gamma_i$ (resp. in $\bigcup_i C_i$).

Note that in general the distributivity condition in the definition of a Grothendieck topology is not satisfied, as shown by the following example (for the sake of simplicity we do not consider the actions):

Example 1. Let $\Sigma = \{0, 1, \leq\}$ and $M = \{0, 1\}$ (with $0 \leq 1$). Consider the following systems: $S_1 = (\Sigma, \{a, b, d\}, \{a \leq b\}, M)$, $S_2 = (\Sigma, \{b, c, d\}, \{b \leq c, c \leq d\}, M)$, $T = (\Sigma, \{a, c\}, \{a \leq c\}, M\}$. Let S the system obtained by interconnecting S_1 and S_2. $S = (\Sigma, \{a, b, c, d\}, \{a \leq b, b \leq c, c \leq d\}, M)$. It is easy to see that in S also $a \leq d$ is satisfied (it is a consequence of the constraints $\{a \leq b, b \leq c, c \leq d\}$. Hence in S_1 seen as a "part" of S, also $a \leq d$ has to be satisfied (by interconnecting systems new constraints may arise). The family $\{S_1, S_2, S_1 \cap S_2\}$ is a cover for S. T is a subsystem of S, but $S_1 \cap T = \{\Sigma, \{a\}, \emptyset, M)$ and $S_2 \cap T = \{\Sigma, \{c\}, \emptyset, M)$, hence $\{S_1 \cap T, S_2 \cap T, S_1 \cap S_2 \cap T\}$ does not cover T.

Therefore, we additionally impose the condition that for every system $S \in \mathsf{InSys}$, the set Γ_S of constraints on the values of the control variables (respectively the set C of constraints that show which parallel actions are admissible), are the restriction of the set Γ_{S_U} (resp. C_{S_U}) to the language of S.

Lemma 5. *The function K assigning for every system S the set $K(S)$ of all covering families for S is a basis (for a Grothendieck topology) on* Sys.

We will denote by J the Grothendieck topology generated by the basis K.

3.1 States and Actions

The states of a system $S = (\Sigma, X, \Gamma, M, A, C)$ are those interpretations $s : X \to M$ that satisfy all the formulas in Γ.

One can see that if $\{S_i \hookrightarrow S \mid i \in I\}$ is a covering site for S, then for every family $\{s_i : X_i \to M_i\}_{i \in I}$ such that for every $i \in I$, $s_i \models \Gamma_i$ and for every $i, j \in I$, $s_{i|X_i \cap X_j} = s_{j|X_i \cap X_j}$ there is a unique $s : X \to M$ such that $s_{|X_i} = s_i$. If, additionally, the covering relation defines a Grothendieck topology on Sys, the following theorem holds.

Proposition 6. *The functor $St : \mathsf{Sys}^{op} \to \mathsf{Sets}$, defined for every object $S = (\Sigma, X, \Gamma, M, A, C)$ in Sys by $St(S) = \{s : X \to M \mid s \text{ satisfies } \Gamma\}$, and for every morphism $S_1 \hookrightarrow S_2$ by $St(S_1 \hookrightarrow S_2)(s) = s_{|P_{S_1}}$ is a sheaf for J.*

The set of admissible actions of the system $S = (\Sigma, X, \Gamma, M, A, C)$ will be the set $Act(S) = \{f : A \to \{0, 1\} \mid f \text{ satisfies } C\}$. We assume that the compatibility of the actions is expressed by the following fact: If $f \in Act(S)$ and if $s \in St(S)$ such that for every $a \in A$ with $f(a) = 1$ there is a $s'_a \in St(S)_{|X_a}$ such that $(s_{|X_a}, s'_a) \in Tr_a$, then the new local states "agree on intersections", i.e. for every $x \in X_{a_1} \cap X_{a_2}$, $s'_{a_1}(x) = s'_{a_2}(x)$. Then we can associate a transition relation to f, that shows how the state of the system changes after the action is performed, namely: $Tr_f \subseteq St(S) \times St(S)$, $Tr_f = \{(s_1, s_2) \mid (s_{1|X_a}, s_{2|X_a}) \in Tr_a$ for every a such that $f(a) = 1$, and $s_1(x) = s_2(x)$ if $x \notin \bigcup_{a, f(a)=1} X_a\}$.

As pointed out before, in what follows we will assume that all the constraints imposed on the actions can be expressed by boolean equations (equations in the boolean algebra freely generated by A).

As in the case of states, it is easy to see that if $\{S_i \hookrightarrow S \mid i \in I\}$ is a covering site for S, then for every family $\{f_i : A_i \to \{0,1\}\}_{i \in I}$ such that for every $i \in I$, $f_i \models C_i$ and for every $i, j \in I$, $f_{i|A_i \cap A_j} = f_{j|A_i \cap A_j}$ there is a unique $f : A \to \{0,1\}$ such that $f_{|A_i} = f_i$. If, additionally, the covering relation defines a Grothendieck topology on Sys, the following theorem holds.

Proposition 7. *The functor* $Act : \mathsf{Sys}^{op} \to \mathsf{Sets}$*, defined for every object* $S = (\Sigma, X, \Gamma, M, A, C)$ *in* Sys *by* $Act(S) = \{f : A \to \{0,1\} \mid f$ *satisfies* $C\}$*, and for every morphism* $S_1 \hookrightarrow S_2$ *by* $Act(S_1 \hookrightarrow S_2)(f) = s_{|A_{S_1}}$*, is a sheaf.*

The restriction imposed on the constraints on actions in particular subsumes also the situation in which a dependence relation on the set of actions is given. This approach leads to the study of traces [Die90]. We can model this case by defining the set of constraints as $\{a_1 a_2 = 0 \mid (a_1, a_2) \in D, a_1 \neq a_2\}$, where D is the dependence relation on A.

Note again that not every type of constraints leads to a sheaf condition. Consider for example the situation when $Act(S) = \{f : A \to \{0,1\} \mid f^{-1}(1)$ is finite$\}$. It can be immediately seen that in this case the sheaf condition is not satisfied for infinite covers.

3.2 Applications

Category theory is a very powerful tool, and its language makes it possible to express things in a uniform way. Many important constructions on systems can for instance be modeled by limits or colimits in the category of systems. We will briefly show what meaning the standard limit and colimit constructions have in the category Sys.

Limits: The *product* of two systems is their largest common subsystem; the *pullback* is the largest common subsystem of two systems that are known to be contained in a larger system. Namely, if for $i = 1, 2$ $S_i = (\Sigma_i, X_i, \Gamma_i, M_i, A_i, C_i) \hookrightarrow S = (\Sigma, X, \Gamma, M, A, C)$, then $S_1 \times_S S_2 = (\Sigma_1 \cap \Sigma_2, X_1 \cap X_2, \Gamma_1 \cap \Gamma_2, M_{12}, A_1 \cap A_2, C_1 \cap C_2)$ where M_{12} is the restriction of M to the signature $\Sigma_1 \cap \Sigma_2$. The terminal object of Sys (if exists) is a system that contains all the systems in Sys.

Colimits: Colimits play a special rôle in the study of systems. Actually the system obtained by interconnecting a given family (diagram) of systems can be obtained computing the colimit of the given diagram.

The *coproduct* $S_1 \amalg S_2$ of two systems S_1, S_2 is the smallest system that contains both of them (and such that the two systems are assumed to be independent, i.e. they have no common subsystems); it is easy to see that if

S_1 and S_2 are independent systems then $St(S_1 \amalg S_2) = St(S_1) \times St(S_2)$, and $Act(S_1 \amalg S_2) = Act(S_1) \times Act(S_2)$. The *push-out* $S_1 \amalg_S S_2$ (see diagram 1) represents the system obtained by interconnecting S_1 and S_2 "gluing" them via their common subsystem S.

$$\begin{tikzcd} S \arrow[r] \arrow[d] & S_1 \arrow[d] \\ S_2 \arrow[r] & S_1 \amalg_S S_2 \end{tikzcd} \tag{1}$$

It is easy to see that if S_1 and S_2 have S as a largest common subsystem, then $St(S_1 \amalg_S S_2) = St(S_1) \times_{St(S)} St(S_2)$ and $Act(S_1 \amalg_S S_2) = Act(S_1) \times_{Act(S)} Act(S_2)$. More generally, we have:

Lemma 8. *Let $\{S_i \mid i \in I\}$ be a covering site for S. Then $S = \varinjlim \{S_i \mid i \in I\}$, $St(S) = \varprojlim \{St(S_i) \mid i \in I\}$ and $Act(S) = \varprojlim \{Act(S_i) \mid i \in I\}$.*

Note that the stability property required in the definition of a Grothendieck topology is in fact a "distributivity property" of intersections of systems w.r.t. arbitrary joins. If we assume that the (partially ordered) category of systems has a Heyting algebra structure, two more operations can be described, namely Heyting implication and negation:

Let S_1 and S_2 be two systems. The system $S_1 \Rightarrow S_2$ is the largest element in the family of systems S with the property that $S_1 \cap S$ is a subsystem of S_2.

Let S be a system. The complement of S (denoted by $\neg S$) is the largest system that is independent of S_1 (in the sense that they have no common subsystems). It is easy to see that $S \cap \neg S = \emptyset$ and that in general $\neg\neg S \neq S$.

We finish by briefly pointing out a possible direction of future work. The fact that states as well as actions can be expressed as sheaves on the category of systems w.r.t. a suitable Grothendieck topology J suggests that also transitions might be defined in a "generic" way as natural transformations between appropriate sheaves. Namely, we can define a natural transformation $tr : Act \to \Omega^{St \times St}$ as follows: For every $S \in \mathsf{Sys}$ let $tr(S) : Act(S) \to \Omega^{St \times St}(S)$ be defined by $tr(S)(f) = tr_f : y(S) \times (St \times St) \to \Omega$, where for every $S' \hookrightarrow S$, $tr_f(S') : St(S') \times St(S') \to \Omega(S')$, with $tr_f(S')(s_1, s_2) = \{S'' \hookrightarrow S' \mid (s_{1|P_{f_{|S''}}}, s_{2|P_{f_{|S''}}}) \in Tr_{f_{|S''}}\}$.

In [Sof96] it is proved that tr_f is a natural transformation for every system S and every action $f \in Act(S)$ and that the functor $tr : Act \to \Omega^{St \times St}$ is a natural transformation. Hence, one can associate with the category of systems Sys a "generic transition system" consisting of (St, Act, tr). It seems (see [Sof96]) that this fact can be exploited in order to study which properties of a system are "inherited" by the building blocks (seen as parts of the interconnection) and which properties of the "building blocks" that compose a system are preserved after their interconnection (possibly using the internal logic in a topos, e.g. $\mathsf{Sh}(\mathsf{Sys}, \mathsf{J})$). In order to achieve this, it is necessary to translate properties of systems (as for example the existence of deadlock, deadlock freedom, determinism) in the internal logic of $\mathsf{Sh(Sys,\ J)}$.

4 Behavior

The starting point of our approach to deal with temporal behavior is the formalism developed by J. Goguen in [Gog92]. He starts with the assumption that every system can be described by a set of attributes X, each attribute $x \in X$ having a prescribed set of values V_x. Assume that time is considered to be discrete. Then in [Gog92] the behavior of a given system S in time is modeled by a functor $F : \mathcal{T}^{op} \to Set$, where $\mathcal{T}$ is the basis for the topology on $\mathbb{N}$ consisting of all the sets $\{0, 1, \ldots, n\}, n \in \mathbb{N}$. Intuitively, for every open set $U = \{0, 1, \ldots, n\}$, $F(U)$ represents the "observations" in the interval of time U. Formally, the functor F is defined on objects by $F(U) = \{h : U \to \prod_{p \in P} V_p \mid K(h)\}$ where $K(h)$ represents a set of conditions that have to be satisfied by h – usually some prescribed rules indicating how the states of the system can change, reflecting the pre- and postconditions of some actions. It is defined on morphisms by $F(\iota_V^U)(h) = h_{|V}$ for every $\iota_V^U : V \hookrightarrow U$ and every $h \in F(U)$. In order to study the behavior of a system consisting of several subsystems the intercommunication between the subsystems is taken into account. A system is seen as a diagram of subsystems, where the morphisms represent inheritance. Goguen shows that the behavior of the system can be described by $F(U) = \{\{h_i \mid i \in I\} \mid h_i \in F_i(U) \text{ and if } \phi_e : S_i \to S_{i'} \text{ then } \phi_e(h_i) = h_{i'}\}$, where for every $i \in I$, F_i is a sheaf that describes the behavior of the system S_i. Therefore, the behavior of a system is the limit of the behaviors of its subsystems (for details see [Gog92]).

In what follows we will develop this idea, modifying the definition for the behavior of a system slightly, by also taking into account the actions that are performed at every step. We will assume that all actions need one unit of time. In future work the more realistic case where actions can have different durations will be considered.

Definition 9. Let S be a system in Sys. The behavior of S is a functor $B_S : \mathcal{T}^{op} \to \mathsf{Sets}$ defined for every $U \in \mathcal{T}$ by $B_S(U) = \{h : U \to St(S) \times Act(S) \mid K(h, U)\}$, where $K(h, U)$ can be expressed by "for every n, if $n, n+1 \in U$ and $h(n) = (s, f)$, then $h(n+1) = (s', f')$ such that $(s, s') \in Tr_f$ and f' can be executed at s'", and for every $\iota : V \subseteq U$, $B_S(\iota) : B_S(U) \to B_S(V)$ is the restriction to V.

Thanks to the particular form of the open sets of $\mathcal{T}$ (all $\{0, 1, \ldots, n\}$ for some n), it can easily be shown that B_S is a sheaf.

Assumption 1 *We assume that if S_1 is a subsystem of S_2, then for every action $a \in A_{S_1}$, if the set $X_a \subseteq X_2$ of variables a depends upon is such that $X_a \cap X_1 \neq \emptyset$, then $a \in A_{S_1}$, and Tr_a^1 is the restriction of Tr_a^2 to variables in $X_a \cap X_1$.*

Assumption 1 says that every action in S_2 that changes some control variables in a subsystem S_1 of S_2 is "known" in S_1, and that its transition relation is the restriction of the transition relation in S_2 to the variables in S_1. In what follows we will assume that Assumption 1 is satisfied.

Let $\iota : S_1 \hookrightarrow S_2$ in Sys. We define $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ by $\rho^{S_2}_{S_1}(U) : B_{S_2}(U) \to B_{S_1}(U)$ for every $U \in \mathcal{T}$, where for every $h : U \to St(S_2) \times Act(S_2)$, $\rho^{S_2}_{S_1}(U)(h) = \langle St(\iota), Act(\iota)\rangle \circ h : U \to St(S_1) \times Act(S_1)$ (with $St(\iota)(s) = s_{|P_1}$ for every $s \in St(S_2)$ and $Act(\iota)(f) = f_{|A_1}$). In what follows, for every $U \in \mathcal{T}$ and every $h \in B_{S_1}(U)$, we will abbreviate $\rho^{S_2}_{S_1}(U)(h)$ by $h_{|S_1}$.

Lemma 10. *Let $U \in \mathcal{T}$ be arbitrary but fixed. Let $B'_U : \mathsf{Sys}^{op} \to \mathsf{Sets}$ be defined for every object $S \in \mathsf{Sys}$ by $B'_U(S) = B_S(U)$ and for every morphism $\iota : S_1 \hookrightarrow S_2$ by $B'_U(\iota) = \rho^{S_2}_{S_1} : B_{S_2}(U) \to B_{S_1}(U)$. Then B'_U is a sheaf.*

Proposition 11. *Let $B : \mathsf{Sys}^{op} \to \mathsf{Sh}(\mathcal{T})$ be defined for every object S of Sys by $B(S) = B_S : \mathcal{T}^{op} \to \mathsf{Sets}$, and for every morphism $\iota : S_1 \hookrightarrow S_2$ by $B(i) = \rho^{S_2}_{S_1} : B(S_1) \to B(S_1)$, and let $B' : \mathcal{T}^{op} \to \mathsf{Sh}(\mathsf{Sys})$ be defined for every $U \in \mathcal{T}$ by $B'(U) : \mathsf{Sys}^{op} \to \mathsf{Sets}$, $B'(U)(S) = B_S(U)$. Then B and B' are functors.*

Proof: By Assumption 1 it follows that for every $h \in B_{S_2}(U)$, $\rho^{S_2}_{S_1}(h) \in B_{S_1}(U)$, i.e. that $\rho^{S_2}_{S_1}$ is well-defined. It is easy to see that $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ is a natural transformation. Let $V \subseteq U$, and let i^U_V be the inclusion of V in U. Then the following diagram commutes:

$$\begin{tikzcd} B_{S_2}(U) \arrow[r, "\rho^{S_2}_{S_1}(U)"] \arrow[d, "B_{S_2}(i^U_V)"'] & B_{S_1}(U) \arrow[d, "B_{S_1}(i^U_V)"] \\ B_{S_2}(V) \arrow[r, "\rho^{S_2}_{S_1}(V)"'] & B_{S_1}(V) \end{tikzcd} \tag{2}$$

Hence B is a functor. In order to show that B' is a functor, note first that from Lemma 10 it follows that B' is well-defined on objects. Let $i : V \hookrightarrow U$ be the inclusion between the open sets $U, V \in \mathcal{T}$. Let us define $B'(i) : B'(U) \to B'(V)$ by $B'(i)(S) : B_S(U) \to B_S(V)$ by $B'(i)(S)(h) = h_U$ for every $h : U \to St(S) \times Act(S) \in B_S(U)$. $B'(i)$ is a natural transformation between the sheaves $B'(U)$ and $B'(V)$. This follows from the commutativity of diagram 2. $\square$

5 Conclusions and Prospects of Future Work

In this paper we made the first steps towards a sheaf-theoretic approach to the study of cooperative systems. We illustrated on a simple example the main ideas that motivated our study. Starting from this example we proposed a definition for systems and we defined a category of systems. We endowed this category with a Grothendieck topology and we showed that in certain (non-restrictive) conditions the states and the (parallel) actions are sheaves with respect to this topology. We used these results for studying the behavior of systems in time.

Of course, this work is only a beginning. One of our next goals is to see which properties can be expressed using results of topos theory: in particular to use geometric logic and to exploit the internal logic in topoi. This is part of ongoing work, see [Sof96]. Another direction for future research is to consider space- and time-dependent formulas. In [PS95] one section is devoted to a concept of space and time dependent formula handling, intended for applications in robotics where the state space of an agent can change with space and time. This makes it possible to model logically a variety of practical situations in the case of cooperating agents working in a geometrically (or topologically) modeled environment X.

Acknowledgments: I thank J. Pfalzgraf for encouraging me to pursue this research and for many stimulating discussions and helpful suggestions. I also thank K. Stokkermans and U. Sigmund for the interesting discussions that made me gain more insight in the field of modeling and planning for cooperating agents.

References

[Die90] V. Diekert. *Combinatorics on Traces*, volume 454 of *Lecture Notes in Computer Science*. Springer Verlag, 1990.

[FS79] M. P. Fourman and D. S. Scott. Sheaves and Logic. In M. Fourman, editor, *Durham Proceedings (1977). Applications of Sheaves*, volume 753 of *Lecture Notes in Mathematics*, pages 302–401. Springer Verlag, 1979.

[Gog75] J. Goguen. Objects. *International Journal of General Systems*, 1:237–243, 1975.

[Gog92] J. Goguen. Sheaf Semantics for Concurrent Interacting Objects. *Mathematical Structures in Computer Science*, 11:159–191, 1992.

[KC79] P. H. Krauss and D. M. Clark. Global Subdirect Products. *Memoirs of the American Mathematical Society*, 17(210):1–109, January 1979.

[Lil93] J. Lilius. A Sheaf Semantics for Petri Nets. Technical Report A23, Dept. of Computer Science, Helsinki University of Technology, 1993.

[Mal94] G. Malcolm. Interconnections of Object Specifications. In R. Wieringa and R. Feenstra, editors, *Working Papers of the International Workshop on Information Systems – Correctness and Reusablity*, 1994. Appeared as internal report IR-357 of the Vrije Universiteit Amsterdam.

[MLM92] S. Mac Lane and I. Moerdijk. *Sheaves in Geometry and Logic*. Universitext. Springer Verlag, 1992.

[MP86] L. Monteiro and F. Pereira. A Sheaf Theoretic Model for Concurrency. *Proc. Logic in Computer Science (LICS'86)*, 1986.

[Pfa91] J. Pfalzgraf. Logical Fiberings and Polycontextural Systems. In P. Jorrand and J. Kelemen, editors, *Proc. Fundamentals of Artificial Intelligence Research*, volume 535 of *Lecture Notes in Computer Science (subseries LNAI)*, pages 170–184. Springer Verlag, 1991.

[Pfa93] J. Pfalzgraf. On Mathematical Modeling in Robotics. In J. Calmet and J.A. Campbell, editors, *AI and Symbolic Mathematical Computing. Proceedings*

AISMC-1, volume 737 of *Lecture Notes in Computer Science*, pages 116–132. Springer Verlag, 1993.

[Pfa95] J. Pfalzgraf. On Geometric and Topological Reasoning in Robotics. *Annals of Mathematics and AI, special issue on AI and Symbolic Mathematical Computing*, 1995. To appear.

[PS95] J. Pfalzgraf and K. Stokkermans. On Robotics Scenarios and Modeling with Fibered Structures. In J. Pfalzgraf and D. Wang, editors, Springer Series Texts and Monographs in Symbolic Computation, Automated Practical Reasoning: Algebraic Approaches, pages 53–80. Springer Verlag, 1995.

[PSS95] Jochen Pfalzgraf, Ute Cornelia Sigmund, and Karel Stokkermans. Modeling cooperative agents scenarios by deductive planning methods and logical fiberings. In Jacques Calmet and John A. Campbell, editors, *2nd Workshop on Artificial Intelligence and Symbolic Mathematical Computing*, volume 958 of *Lecture Notes in Computer Science*, pages 167–190. Springer-Verlag, 1995.

[PSS96] J. Pfalzgraf, U. Sigmund, and K. Stokkermans. Towards a General Approach for Modeling Actions and Change in Cooperating Agents Scenarios. *IGPL (Journal of the Interest Group in Pure and Applied Logics)*, 4(3):445–472, 1996.

[Sof96] V. Sofronie. A sheaf theoretic approach to cooperating agents scenarios. Manuscript in preparation, 1996.

[Sri93] Y.V. Srinivas. A Sheaf-theoretic Approach to Pattern Matching and Related Problems. *Theoretical Computer Science*, 112:53–97, 1993.