Electronic Notes in Theoretical Computer Science 48 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume48.html pp. 1 – 16

Constraint Based Coordination of Autonomous Agents

Sascha Ossowski [1]

Artificial Intelligence Group, E.S.C.E.T.
University Rey Juan Carlos
Madrid, Spain

Abstract

This paper presents an instrumentation of a constructionist bottom-up approach to coordination in multiagent systems. Coordination scenarios of autonomous agents are modelled and formalised as a contraint optimisation problem. A distributed asynchronous algorithm is presented that computes potential outcomes of coordination in these scenarios. Finally, it is shown how this algorithm can be integrated into an operational coordination mechanism for a real-world domain.

1 Introduction

The field of Multiagent Systems (MAS) is concerned with the study of systems of multiple interacting artificial agents. Coordination provides the "control structures" in such a system, so it does not surprise that it is one of the major topics of MAS research. It is being tackled from different points of interest: besides Social Simulation [3] and Mechanism Design [8], the efficient construction of agent-based *problem-solving* systems is a primary objective. In these systems, a single designer creates a purposefully designed architecture of computational agents that interact in order to achieve jointly a desired global functionality.

In this context, the traditional design philosophy is *reductionism* [6]. It relies on a top-down decomposition of the global task, the assignment of subtasks to agents and coordination based on pre-established interaction patterns among *benevolent* agents. In these systems, agents have to account for interdependencies between their local behaviours in the light of control uncertainty: they have to find coherent sets of local behaviours despite incomplete and potentially inconsistent views of the overall problem-solving state. Coordination problems of these characteristics have often been formalised as *distributed*

[1] Email: s.ossowski@escet.urjc.es

search in the agent behaviour space [2]. A popular model for the instrumentation of coordination is *distributed constraint satisfaction* [10], [11]: the interdependencies between agent behaviours are modelled as constraints, and the messages exchanged among agents are conceived as the messages of a distributed constraint satisfaction algorithm.

However, reductionist approaches to coordination in multiagent problem solving systems often turn out to be too rigid for real-world applications [6]. Instead, a *constructionist* approach, based on the metaphor of societies of autonomous problem-solving agents, has become popular: agents are primarily interested in their local goals and interact to increase the degree of their attainment. In order that the system copes with the global task, a constructionist approach to coordination must be based on a mechanism that biases agent behaviour in a desired direction, i.e. somewhere between benevolence and self-interest. Elsewhere [7], we described the fundamentals of *structural cooperation*, an example of a coordination mechanism of this type.

In this paper, we present an algorithm for the instrumentation of structural cooperation based on distributed constraint-based search and illustrate its application to a particular domain. Section 2 characterises the coordination scenarios that we are interested in. In Section 3, we present a distributed constraint optimisation algorithm, that computes potential outcomes of coordination in these scenarios. In Section 4 we point to a real-world application of this coordination approach, before presenting our conclusions in Section 5.

2 The problem

We first provide a formal description of the type of multiagent domains that our coordination scenarios are based on: systems of cognitive agents that *reactively* develop short-term plans. Elsewhere [1], we have argued that such a stance is appropriate for a variety of real world scenarios, such as many decisions support domains. Section 2.2 gives a simplified example of one of the targeted domains.

2.1 The coordination setting

Let S be a set of world states and Π a finite set of plans. The execution of a plan π changes the state of the world, which is modelled as a partially defined mapping

$$res : \Pi \times S \rightarrow S$$

A plan π is executable in s, if only if res is defined for a certain world state s. We express this formally by means of the predicate $exec(\pi, s)$. There is a set of agents A, each of which can act in the world thereby modifying its state.

Definition 2.1 An agent $\alpha \in A$ is characterised by the following notions: [2]

- a *preference relation* $\precsim_\alpha$ over world states;
- a set of *legally enactable individual plans* $\Pi^s_\alpha \subseteq \Pi$, where $res(\pi, s)$ is defined for all $\pi \in \Pi^s_\alpha$. [3]

Legally enactable individual plans represent an agent α's action alternatives, while the preference relation $\precsim_\alpha$ expresses α's desires respecting world states to bring about.

We now introduce a notion of simultaneous and interdependent action. The set M of *multiplans* comprises all multisets over the individual plans Π, i.e. $M = bagof(\Pi)$. A multiplan $\mu \in M$ models the simultaneous execution of all its component plans, i.e. of the individual plans $\pi \in \mu$ that are contained in the multiset μ. The commutative operator $\circ$ denotes multiset union and hence states that its operands are to be executed together. [4]

By identifying an individual plan with a multiplan that contains it as its only element, the partial function res is extended to multiplans:

$$res : M \times S \rightarrow S$$

The function res is undefined for a multiplan μ and a state s, if some of the individual plans that it contains are incompatible, i.e. in case that in a state s of a modelled domain it is impossible to execute them simultaneously. Otherwise, μ is said to be executable in s (formally: $exec(\mu, s)$).

We define the set of *groups* Γ as the powerset of the set of agents, i.e. $\Gamma = \wp(A)$. An *assignment* is a bijective function Ψ that maps each group member $\alpha \in \gamma$ to exactly one of the individual plans $\pi \in \mu$ in a multiplan μ. Note that an assignment always relates n agents to a multiplan μ comprising n individual plans, even though μ contains some individual plans more than once.

We can then extend the notion of legally enactable plans to groups:

Definition 2.2 The set of *legally enactable multiplans* $M^s_\gamma \subseteq M$ contains all $\mu \in M$ such that

- $res(\mu, s)$ is defined, i.e. μ is executable;

[2] The definitions given in this section are expressed with respect to a state s. For the purpose of illustrating the coordination algorithm to be presented in Section 3, we can assume s to be a fixed "current" state.

[3] The mechanism of structural cooperation actually requires that a legally enactable plan π of agent α be executable in a state s ($exec(\pi, s)$), that α be capable of enacting π ($can(\alpha, \pi)$), and that it not be prohibited for α to execute π ($\neg forbidden_s(\alpha, \pi)$). The set may change with s, either because the new situation modified the executability of some plan, or because the system designer chose to modify the prohibitions respecting some π. However, as indicated above, for the sake of this paper the "higher level" notion of legally enactable plans in a current state s is sufficient.

[4] In the sequel we will use a set of cardinality one and its only element indiscriminately. So, for instance, we write and $\mu = \pi' \circ \pi = \pi \circ \pi' = \{\pi, \pi'\}$ and $\mu \circ \pi = \pi \circ \mu = \{\pi, \pi, \pi'\}$

- there is an assignment Ψ such that $\forall \alpha \in \gamma.\, \Psi(\alpha) \in \Pi^s_\alpha$. [5]

The coordination settings that we are interested in are determined by the sets of individuals S, Π, A and the function res, as well as for each agent α the predicate $\precsim_\alpha$ and the set Π^s_α. We are particularly interested in a specific kind of coordination settings: a coordination setting is *detached*, if an agent's acquaintances may influence the executability of its individual plans (the truth value of the predicate *exec*), but not its preference respecting their outcome (the state resulting from the function res).

Definition 2.3 A coordination setting is *detached*, if for all agents $\alpha \in A$ and all groups $\gamma \in \Gamma$, $\alpha \in \gamma$ holds: if $\pi \in \Pi^s_\alpha$ is a legally enactable plan of α, $\mu \in M$ is a multiplan, and $(\pi \circ \mu) \in M^s_\gamma$ is legally enactable by γ, then
$res(\pi, s) \sim_\alpha res(\pi \circ \mu, s)$

Finally, let us have a short look at potential outcomes of a coordination process in such a scenario. Obviously, this outcome determines the individual plans that every agent chooses to enact, and can thus be modelled as a multiplan. Given the fact that we are interested in coordination among autonomous (self-interested) agents, several requirement for the coordination process and its outcome can be postulated.

Firstly, a solution needs to be *individually rational*: the agents will converge on an agreement if they are better off coordinating with their acquaintances than individually. Still, in almost every interesting multiagent domain this is the case. [6]

More important, the coordination outcome is to be "efficient". Let μ and μ' be two legally enactable multiplans. We say that μ *dominates* μ' if the result of executing μ is weakly preferred by all agents to the result of executing μ. This is

$$\forall \alpha \in A.\, res(\mu', s) \precsim_\alpha res(\mu, s)$$

A legally enactable multiplan μ is Pareto-optimal, if it is undominated by any other legally enactable multiplan. This is to say that in all other legally enactable multiplans μ', there is at least one agent α that strongly prefers the result of μ to the outcome of μ'.

Definition 2.4 A legally enactable multiplan $\mu \in M^s_\gamma$ is *Pareto-optimal* if
$\forall \mu' \in M^s_A, \mu' \neq \mu, \exists \alpha \in A.\ res(\mu', s) \prec_\alpha res(\mu, s)$

[5] Structural cooperation does not actually require $res(\pi, s)$ to be defined for all $\pi \in \mu$. So, it considers the case that a certain π may not be executable alone, but that other plans $\pi' \in \mu$ contain actions that fill the "gap" in π and thus make it executable as part of the multiplan μ. However, throughout this paper, we stick to the stronger assumption that all $\pi \in \mu$ of a legally enactable multiplan μ also have to be legally enactable.

[6] One of the few exceptions are two-player zero-sum games, where the agents' interests are totally antagonistic.

Fig. 1. Distributed Traffic Management Scenario

Pareto-optimality assures that no option to improve (or at least: not to deteriorate) an agreement in the eyes of all agents is wasted. This is particularly relevant for multiagent problem-solving systems, where we can assume that the local preference relations of the agents correlate positively with the desired global functionality of the system. Still, there are usually several Pareto-optimal multiplans. It is the particular coordination mechanism that establishes how to choose among them.

2.2 An Example: the traffic management domain

As an example of the coordination settings that we are targeting, consider a system in charge of managing urban road traffic, so as to maintain and restore the "smooth" flow of vehicles. Such a system receives information about the current traffic state (usually by means of "loop detectors" installed under the road surface), and generates *signal plans* to influence traffic flows and alleviate problems. Signal plans determine the state of traffic control devices, such as the messages to be shown on Variable Message Signals (VMS) installed above the road, or cycle times of traffic lights at junctions.

A distributed approach to the design of such traffic management systems has many advantages [1]. Setting out from the traffic engineers' knowledge respecting the usefulness of a logical subdivision of the road network into *problem areas*, we can build a distributed system that relies on a set of autonomous traffic control agents, each responsible for the traffic management in one such areas (see Figure 1). Once an agent detects traffic problems in its area, it generates a set of alternative local signal plans to overcome them, and ranks them according to their estimated effectivity with respect to its local traffic problems. Still, local signal plans are interdependent: they may either be *incompatible* and use the same control device in incompatible ways (e.g. display different messages on the same VMS), or they may be *interfering*, changing traffic flows not only in their local but also in neighbouring areas (thus modifying the effectivity of other agents' plans) [4].

The problem of coordinating these agents can be modelled in the above

terms:

- the alternative signal plans for a traffic agent's problem area correspond to the set of legally enactable individual plans.
- the ranking of an agent α's alternative signal plans corresponds to the local preference relation $\precsim_\alpha$ on the resulting world states
- the simultaneous execution of the agents' local signal plans corresponds to a multiplan. Incompatibility of local signal plans means that the corresponding multiplan is not legally enactable. Interference is modelled by the fact that the result of an individual plan π changes when enacted within the multiplan μ.

Consider, for instance, the following situation, where 3 traffic agents α_1, α_2 and α_3 each generate 3 alternative signal plans with the following local preferences over them:

$$\begin{array}{lll} \Pi^s_{\alpha_1} = \{\pi_{a_1}, \pi_{a_2}, \pi_{a_3}\} & \text{with} & \pi_{a_1} \succ_1 \pi_{a_2} \succ_1 \pi_{a_3} \ , \\ \Pi^s_{\alpha_2} = \{\pi_{b_1}, \pi_{b_2}, \pi_{b_3}\} & \text{with} & \pi_{b_1} \succ_2 \pi_{b_2} \succ_2 \pi_{b_3} \ , \\ \Pi^s_{\alpha_3} = \{\pi_{c_1}, \pi_{c_2}, \pi_{c_3}\} & \text{with} & \pi_{c_1} \succ_3 \pi_{c_2} \succ_3 \pi_{c_3} \ . \end{array}$$

In our example, suppose that the following sets of individual plans are incompatible, i.e. do not constitute a legally enactable multiplan for the group $\{\alpha_1, \alpha_2, \alpha_3\}$:

$$\{\pi_{a_1}, \pi_{b_1}, \pi_{c_1}\}, \{\pi_{a_1}, \pi_{b_1}, \pi_{c_2}\}, \{\pi_{a_1}, \pi_{b_2}, \pi_{c_1}\}, \{\pi_{a_1}, \pi_{b_2}, \pi_{c_2}\}, \{\pi_{a_2}, \pi_{b_1}, \pi_{c_1}\},$$
$$\{\pi_{a_2}, \pi_{b_2}, \pi_{c_1}\}, \{\pi_{a_1}, \pi_{b_1}, \pi_{c_3}\}, \{\pi_{a_1}, \pi_{b_3}, \pi_{c_1}\}, \{\pi_{a_3}, \pi_{b_1}, \pi_{c_1}\}, \{\pi_{a_2}, \pi_{b_3}, \pi_{c_1}\}$$

If the subdivision of the road network is defined in such a way that the possible variation of flows between problem areas is negligible, the coordination setting is *detached*: agents only compete for the use of control devices, but their preference over their local signal plans is not affected by the choices of others. In our example we adopt this assumption. So, for instance, agent α_1 is indifferent respecting the following multiplans:

$$\{\pi_{a_1}, \pi_{b_2}, \pi_{c_3}\} \sim_1 \{\pi_{a_1}, \pi_{b_3}, \pi_{c_2}\} \sim_1 \{\pi_{a_1}, \pi_{b_3}, \pi_{c_3}\}$$

In such a setting, it can be easily checked that there is a total of 4 Pareto-optimal legally enactable multiplans:

$$\mu_1 = \{\pi_{a_1}, \pi_{b_2}, \pi_{c_3}\};\ \mu_2 = \{\pi_{a_1}, \pi_{b_3}, \pi_{c_2}\};\ \mu_3 = \{\pi_{a_2}, \pi_{b_1}, \pi_{c_2}\} \text{ and}$$
$$\mu_4 = \{\pi_{a_3}, \pi_{b_2}, \pi_{c_1}\}.$$

3 Distributed Search for Pareto-optimality

In this section we describe our distributed algorithm for computing undominated (i.e. Pareto-optimal) legally enactable multiplans in the aforementioned coordination scenario. After relating the coordination scenario to constraint optimisation, Section 3.2 describes the algorithm in further detail and illustrates its dynamics. An analysis of the algorithm is given in Section 3.3.

3.1 Coordination as constraint optimisation

In this section we model our coordination scenario as a constraint optimisation problem (COP). In COPs each agent has control over a *decision variable* to which it can assign values from its *local domain*. *Constraints* involve two or more agents' variables, determining whether sets of assignments of values to the agents' variables are *consistent* or inconsistent. In addition, there is a preference ordering over value assignments. We call a set of value assignments *consistent*, if it complies with all constraints. It is a *solution*, if it is consistent and also "optimal" with respect to the preference ordering.

Our original problem of finding legally enactable, undominated multiplans in a coordination setting can be mapped onto such a distributed constraint optimisation problem without major problems.

- An agent's decision variable corresponds to the choice of the legally enactable individual plans π that it is going to enact; its local domain is given by the set Π^s_α.
- The constraints between variables are implied by multiplan executability. Consistent sets of value assignments are a synonym for legally enactable multiplans.
- Setting out from the subsequent notion of local preference $\precsim_\alpha$ for each agent α, the preference ordering over legally enactable multiplans is given by multiplan domination.

So, a solution to our constraint optimisation problem (i.e. an "optimal" value assignment) corresponds to a Pareto-optimal legally enactable multiplan. In order to contribute to a solution of coordination scenarios modelled as COPs of these characteristics, each agent α needs to be endowed with the following knowledge:

- its legally enactable individual plans $\pi \in \Pi^s_\alpha$;
- the way that legally enactable multiplans μ of its acquaintances may influence its individual plans $\pi \in \Pi^s_\alpha$, respecting their executability in a situation s ($exec(\pi \circ \mu, s)$) as well as respecting their outcome ($res(\pi \circ \mu, s)$);
- its local preference $\precsim_\alpha$ on world states.

We aim at designing an algorithm that determines possible outcomes of our coordination scenario (Pareto-optimal legally enactable multiplans) by computing solutions to the corresponding COP.

3.2 The Algorithm

In decentralised MAS, coordination should rely on a decentralised mechanism. So, we are interested in an asynchronous, distributed algorithm for computing solutions to the above COP. Such algorithms are different from parallel/distributed methods for constraint satisfaction, in that there is an a priori distribution of problem knowledge among asynchronously acting agents,

while the latter aim to design a distributed architecture in order to generate solutions more efficiently through parallelism [12].

Our particular approach to the computation of such solutions is based on *asynchronous weak commitment search*: a dependency-directed backtracking algorithm adapted to *distributed* constraint satisfaction problems [11]. We have extended this algorithm to cope with the specific characteristics of our constraint *optimisation* scenario.

In the following we first give a short overview of the fundamentals of our algorithm. Subsequently, we describe the information models that each agent maintains during the distributed search process. After having a closer look at the notion of local consistency and examining the message types used in our distributed algorithm, we finally present our agent programs that lead to an asynchronous and distributed computation of the set of undominated legally enactable multiplans.

3.2.1 Overview

The algorithm assumes that each agent follows the same simple *agent program*: in every instant of time, an agent chooses from its legally enactable individual plans the one that results in a current multiplan that is "locally consistent" and most preferred by it. The agents' attempts to achieve local consistency produce a chain reaction, which eventually leads to a stable state. The definition of local consistency in conjunction with this local choice strategy assures that in stable states, when all agents are waiting for messages, their choices of individual plans constitute a legally enactable and hopefully undominated multiplan.

A major difficulty in an asynchronously acting agent system is to avoid cycles, i.e. to prevent infinite processing. This is achieved by dynamically generating *nogoods*. Nogoods are multiplans that cannot be extended to a legally enactable multiplan involving all agents. As such, they describe new, dynamic constraints: all multiplans that constitute a superset of a nogood are not legally enactable.

We are interested in the potential of finding all undominated legally enactable multiplans. Therefore, once a solution has been found, the search process must be restarted until no further solutions are present. For this purpose, previously found solutions are recorded as another type of dynamic constraint. These solution constraints assure that the same legally enactable solutions cannot be encountered twice. In the sequel, we describe the key aspects of the agents programs.

3.2.2 Information models and local consistency

The program of an agent α requires that certain dynamic information involving α be available in a so-called self model:

- its *current value*, i.e. a tentative legally enactable individual plan $\pi \in \Pi_\alpha^s$;

- its current *authority level* $v_\alpha \in \aleph$ (see below);
- the set Ξ of previously generated *nogoods* χ that α is involved in
- the set R of previously found *solutions* ρ that α is involved in

In addition, the following information about other agents is maintained in specific acquaintance models. For each agent $\tilde{\alpha}$ that α shares some constraint with such a model keeps track of

- $\tilde{\alpha}$'s *current value*, i.e. a tentative legally enactable individual plan $\pi \in \Pi^s_{\tilde{\alpha}}$;
- $\tilde{\alpha}$'s current *authority level* $v_{\tilde{\alpha}} \in \aleph$.

By joining the information on the current value from the self model and all acquaintance models, an agent has information respecting the *current multiplan* μ.

Local consistency is defined with relation to an agent's *authority level*, a unique integer value for each agent that defines a total order among them. An individual plan of an agent α is locally consistent, if it is part of a current multiplan that fulfils all constraints involving only *higher* authority agents.

Definition 3.1 Let $\pi \in \Pi^s_\alpha$ and $\mu \in M^s_\gamma$ where γ contains all agents with higher authority than α. Agent α's plan π is *locally consistent*, if the current multiplan $\pi \circ \mu$ obeys the following conditions:

- it complies with all genuine constraints;
- it is not a superset of a nogood;
- it is does not coincide with a solution.

A locally consistent value is called an *option* of the agent. As we will argue later, the authority levels do not affect the result of the distributed search process, but influence the order in which undominated and legally enactable multiplans are found. They are not part of the coordination setting, but a technical means used by the distributed algorithm to compute Pareto-optimal plans within the setting.

3.2.3 Information models and messages

In order to keep their information models up-to-date, agents exchange messages. [7] These are the essential messages and their "semantics": [8]

- *value* messages: the sender informs the receiver that its current value has changed.
- *nogood* messages: the sender informs the receiver that it has detected a nogood.

[7] We assume a communication model with finite but random delay in message delivery, whereas messages are received in the order in which they are sent.

[8] There are some additional "control" messages. For instance, *timeout* messages abort the algorithm immediately, by requiring agents to enter directly into a locally consistent state. By means of this, we can make use of the anytime properties of the algorithm [7].

- *authority* messages: the sender informs the receiver that its current authority level has changed.
- *solution* messages: the sender informs the receiver that it has detected a solution.

Upon the reception of these messages, the receivers update their self or acquaintance models accordingly. If a corresponding acquaintance model does not yet exist, it is created.

Once an agent updates its information models as a result of the choices implied by its agent program (see next section), relevant acquaintances are informed through messages. If an agent updates its own current individual plan (or: its own authority level), *value* (or: *authority*) messages are sent to all agents that it shares some constraints with. In case that a nogood has been detected, all agents that may be involved in it will receive *nogood* messages. *Solution* messages are targeted to all agents.

3.2.4 Agent programs

We are now provided with all ingredients to describe the dynamics of agent programs. The agent that first detects a new situation s initiates the distributed algorithm. This *leader agent* plays a special role[9]: It is in charge of detecting solutions, as well as of collecting the corresponding legally enactable multiplans. Apart from that, it behaves like all other agents.

Once an agent α receives a message from an acquaintance, it updates its information models accordingly and calculates its options, i.e. the set of legally enactable individual plans that are locally consistent with respect to its information models. Subsequently, if it detects to be in a locally inconsistent state, it behaves according to the following local choice strategy.

```
METHOD localChoice (Options)
        IF Options = ∅ THEN
                Nogoods←calcNewNogoods()
                FOR EACH μ ∈Nogoods DO
                        updateModels(nogood, μ)
                v' ←determineAuthority()
                IF selfModel.authority ≠ v' THEN
                        updateModels(authority, v')
                        NewOptions←calcNewOptions(v')
                        localChoice(NewOptions)
        ELSE // Options ≠ ∅
                π' ← max_{≼α} {π ∈ Options}
                updateModels(value, π')
```

[9] Note that every agent can initiate the interaction process, i.e. every agent can play the role of the leader.

```
END METHOD
```

Given the example 3-agent coordination setting from Section 2, suppose that α_1 has the highest authority level, followed by α_2 and α_3. Furthermore, assume that α_2's current value is π_{b_1}, and that α_3's current value is π_{c_1}. Now, both agents receive *value* messages from α_1, the leader, indicating that its current value has changed to π_{a_1}.

Upon reception of the *value* messages, α_2 and α_3 update their acquaintance models of α_1 accordingly. As it only needs to take into account constraints with α_1, agent α_2 remains in a locally consistent state and does not react. Still, α_3's current value is not locally consistent. Moreover, its set of options is also empty (there is no way to build a locally consistent multiplan with the current values of α_1 and α_2): asynchronous search has come to a dead-end and backtracking needs to be done. For each subset of the current multiplan that makes it impossible to restore local consistency, i.e. for each combination of values of higher authority agents that leaves no options to the agent, a nogood is generated and models are updated accordingly. In our example, α_3 computes the nogood $\{\pi_{a_1}, \pi_{b_1}\}$, and sends the corresponding *nogood*-messages to α_1 and α_2 (the agents involved in the nogood). Suppose that *determineAuthority* settles a kind of distributed search that does not require a change of authority of α_3 at this time [10].

Upon reception of the *nogood* messages, α_1 and α_2 update their self models. α_1 remains locally consistent, as it is not the lowest authority agent involved in the nogood $\{\pi_{a_1}, \pi_{b_1}\}$. Still, α_2 is, and so it calculates its set of options in order to restore its local consistency. Option comprise π_{b_2} and π_{b_3}, and the locally most preferred π_{b_2} is chosen. α_2 updates its self model accordingly, and all related agents are informed about the value change. In the example, *value* messages are sent to α_1 and α_3.

Upon reception of the *value* messages, α_1 and α_3 update their acquaintance models of α_2. Again, α_1 is still locally consistent, but α_3 needs to calculate its set of options, which is now limited to only π_{c_3}. The self model is updated and the corresponding *value* messages are sent to α_1 and α_2. When agents α_1 and α_2 receive these messages, they update their acquaintance models of α_3 accordingly, but both remain in a locally consistent state.

Now, the leader agent α_1 detects that all agents are in a stable, locally consistent state [11]. By consequence, a solution has been found and the leader

10

sends *solution* messages to all acquaintances. Upon the reception of these messages, agents store the solution in their self models. In the example, agent α_2 and α_3 record the Pareto-optimal multiplan $\{\pi_{a_1}, \pi_{b_2}, \pi_{c_3}\}$ as a new solution constraint.

Still, due this new constraint agent α_3 has come into a locally inconsistent state. So, the distributed search process recommences, but with a search space of reduced size. The algorithm concludes when no more legally enactable multiplans can be found. This is detected when an empty nogood is generated and the method *updateModels* tries to store it in the self model.

3.3 Analysis

The above algorithm has been designed to compute potential outcomes of coordination in decentralised multiagent problem-solving systems. This type of systems is usually applied to domains that show an *a priori* distribution: the subdivision of the problem-solving system into different agents and the corresponding agent processes is rather the result of a given *problem structure* (spatial distribution, access restrictions etc. [7]), than a means to attack computational complexity issues. Upon this background, in this section we analyse some of the properties of the aforementioned algorithm.

The different agent processes perform parallel, asynchronous computation: *nogood* messages only imply the reaction of one agent (the one with the lowest authority level in the nogood), but messages informing about value changes cause simultaneous activity. The communication load is limited by selective addressing. Note that when acquaintances are informed about new nogoods, it would suffice to direct the corresponding messages just to the lowest authority agent involved in the nogood. Still, authority levels are changing and our policy of routing nogood messages anticipates potential future communication. Furthermore, the search space is pruned on the basis of nogoods, as all multiplans that comprise a nogood are not explored.

Respecting soundness and completeness of the algorithm, we have obtained the following results.

Theorem 3.2 *Suppose a coordination setting, where all agents apply the agent program. The following holds: after finite time the algorithm terminates and the leader is endowed with a superset of all undominated, legally enactable multiplans.*

Corollary 3.3 *In a detached coordination setting, the leader hosts precisely the set of undominated, legally enactable multiplans.*

Proofs can be found in [7]. Another interesting question concerns the complexity of the distributed algorithm. As the whole search space needs to be

---

the leader with such a snapshot. Details on this solution detection method can be found in [7].

traversed in order to guarantee completeness, the algorithm's time complexity is exponential in the number of agents in the worst case. This is underlined by the NP-completeness of the constraint problems, that the problem of finding legally enactable multiplans has been mapped onto.[12]

The worst case space complexity for each agent is determined by the number of recorded nogoods. So, it may also grow exponentially in the number of agents. This seems inevitable as both completeness and a dynamic traversal of the space of multiplans have to be assured.

Still, despite these "unpleasant" complexity results, the algorithm is effectively applicable to coordination scenarios with relatively few agents and small sets of legally enactable individual plans.

4 An Application

In this section we outline how the above algorithm can be applied to a real-world problem. It is part of the instrumentation of the coordination mechanism called structural cooperation, which as been used in the TRYSA_2 prototype (TRYS Autonomous Agents) for urban traffic management [7][4]. TRYSA_2 consists of 11 autonomous knowledge-based traffic control agents, that jointly manage a road network consisting of one ring-road and seven adjacent motorways, provided with over 300 loop detectors, 52 Variable Message Signals (VMS), 3 traffic lights for junction control, as well as ramp metering on 7 ring-road drives.

Agents are provided with knowledge about the dynamics of traffic flows in their particular problem area and use advanced reasoning techniques to come up with alternative local signal plans to overcome them. Once a traffic problem is detected by an agent, it determines its magnitude in terms of the amount of traffic demand that exceeds the capacity of the area's road segments. In much the same way, alternative signal plans are ranked by their *expected reduction in traffic excess*, i.e. a signal plan is the more preferred that more it is supposed to reduce traffic excess in the problem area. We call the expected reduction of traffic excess by a signal plan its local *utility* for the corresponding traffic agent.

The mechanism of structural cooperation makes use of this additional information on plan utility when determining the outcome of coordination.

Firstly, it allows the agents to agree on a *gamble* respecting which multiplan to enact: they may toss a coin and, according to the outcome of this experiment, execute one or another multiplan. This gamble is referred to as *mixed multiplan*. The utility of a mixed multiplan is the weighed sum of the utilities of the (pure) multiplans that it comprises. Second, besides individual rationality and Pareto-optimality, structural cooperation requires a coordina-

[12] Given these findings, in some settings it will be necessary to rely on the anytime properties of the algorithm (or to limit in advance the number of solutions to be searched for).

tion outcome to be symmetric (no agent has an a priori advantage), invariant under certain types of linear transformations of the utility function, and independent of irrelevant alternatives (additional "bad" plans of an agent do not influence the outcome of coordination). Under these conditions, it has been shown that autonomous (self-interested) agents will agree on the mixed multiplan, that maximises the product of gains for each agent, compared to the situation of disagreement where every agent tries to cope with its local problems without taking into account its acquaintances. This is called the Nash bargaining solution [9].

The above finding is particularly interesting, because it allows us to compute the outcome of the coordination process of autonomous agents by means of an algorithm in which agents behave benevolently. In particular, the independence of irrelevant alternatives assures that agents only gamble on Pareto-optimal plans. This allows for the integration of the distributed algorithm described in Section 3 into the multistage coordination algorithm of TRYSA$_2$. As Figure 2 depicts, according to this algorithm the coordination process is a sequence of three stages.

- Stage 1: asynchronous search for Pareto-optimality
  the asynchronous distributed constraint optimisation algorithm of Section 3 determines undominated, legally enactable multiplans
- Stage 2: determination of the Nash bargaining solution
  the agent that detects the termination of Stage 1 plays the role of the leader in this stage. On the basis of the result of Stage 1, it computes the (approximate) product-maximising solution in mixed multiplans.
- Stage 3: probabilistic assignment of individual plans
  the leading agent generates a lottery in accordance with the outcome of Stage 2, and urges its acquaintances to execute the corresponding individual plans (i.e. local signal plans) accordingly.

The TRYSA$_2$ system has been implemented experimentally on networked workstations. The TRYSA agents constitute separate Prolog processes (with some extensions in C++), which communicate via sockets. The road network is simulated by a traffic simulator that performs microscopic ("car by car") simulation of traffic flows. A special observer agent has been implemented in Tcl/Tk in order to visualise the problem-solving process and its results [7].

5 Conclusions

In this paper we have shown how a coordination scenario of multiple autonomous problem-solving agents can be modelled as a constraint optimisation problem. We have presented and analysed a distributed asynchronous constraint optimisation algorithm to compute the Pareto-optimal solutions of the corresponding coordination scenario. Finally, we have used the traffic management domain to illustrate the instrumentation of a coordination

Fig. 2. Multistage coordination algorithm

mechanism based on this algorithm.

We plan to evaluate the mechanism of structural cooperation, as well as the algorithm that instruments it, in other domains. We are particularly interested in a comparison to centralised solutions, as realised in [5] for the traffic domain. In addition we are working towards a tighter integration of the Stages 1 and 2 of our coordination algorithm in order to improve its computational complexity.

References

[1] Cuena, J. and S. Ossowski, *Distributed models for decision support*, in: G. Weiss, editor, *Multi-agent Systems – A Modern Approach to DAI*, MIT Press: Cambridge, 1999 pp. 459–504.

[2] Durfee, E. H. and T. A. Montgomery, *Coordination as distributed search in a hierarchical behaviour space*, IEEE Transactions on Systems, Man, and Cybernatics, Special Issue on DAI **21** (1991).

[3] Gilbert, N. and R. Conte, "Artificial Societies – The Computer Simulation of Social Life," UCL Press: London, 1995.

[4] Hernandez, J., S. Ossowski and A. Garcia-Serrano, *Agent-based decision support for traffic management – experiences in a real-world scenario*, in: *Agents-2000 Workshop on Autonomous Agents in Traffic Modelling*, 2000, pp. 30–34.

[5] Hernandez, J., S. Ossowski and A. Garcia-Serrano, *Multiagent architectures for intelligent traffic management systems*, to appear in: Transportation Research (Part C) (2001).

[6] Jennings, N. and J. Campos, *Towards a social level characterisation of socially responsible agents*, IEE Proc. on Software Engineering **144** (2001).

[7] Ossowski, S., "Co-ordination in Artificial Agent Societies," Lecture Notes in Computer Science **1535**, Springer-Verlag, Berlin, 1999.

[8] Rosenschein, J. S. and G. Zlotkin, "Rules of Encounter: Designing Conventions for Automated Negotiation among Computers," AI, MIT Press, Cambridge, Mass, 1994.
URL `http://www-sloan.mit.edu/CCS/giladbook.html`

[9] Sandholm, T., *Distributed rational decision making*, in: G. Weiss, editor, *Multi-agent Systems – A Modern Approach to DAI*, MIT Press: Cambridge, 1999 pp. 201–258.

[10] Sycara, K., S. Roth, N. Sadeh and M. Fox, *Distributed constrained heuristic search*, IEEE Transactions on Systems, Man, and Cybernetics **21** (1991), pp. 1446–1461.

[11] Yokoo, M., *Search algorithms for agents*, in: G. Weiss, editor, *Multi-agent Systems – A Modern Approach to DAI*, MIT Press: Cambridge, 1999 pp. 165–200.

[12] Yokoo, M. and K. Hirayama, *Algorithms for distributed constraint satisfaction: A review*, Autonomous Agents and Multi-Agent Systems **3** (2000), pp. 185–207.