1 Introduction, Structure of the Talk

– introduction (this is it)

– ARL project specification

– domains/scenario

– distributed planning

– selected techniques

– tasks for next, discussion

2 ARL sponsored project

title:	Distributed Planning and Coordination of Team-oriented Activities
funded by:	European Research Office – USARDSG-UK
project partners:	ATG at Gerstner Lab, CTU and AIAI at University of Edinburgh
duration:	24 months (secured funding for the first 4 months)

The project is supposed to leverage:

– ATG expertise in the field of multi-agent systems, coordination, handling interaction inaccessibility in ad-hoc networks and implementing computational reflection in dynamic environment.

– AIAI capability and research record in the field of distributed coordination, man-machine interaction in heterogeneous environment, as well as knowledge activity management methods.

– Both centers are skilled in the area of planning.

2.1 Goals of the project

The goal of the project is to provide:

1. distributed planning architecture and collection of different methods of distributed planning and coordination in the specific, goal directed environment.
2. testing and demonstration scenario, computational model and a prototype testbed

The environment specific features will be based mainly on (i) ad-hoc type of partially inaccessible communication infrastructure, (ii) interaction among semi-trusted and in parts collaborative, in parts self-interested actors and (iii) distribution of knowledge required for efficient coordination.

– design of task decomposition and plan merging methods and mechanisms for negotiating about the partial plans,

– methods for planning with incomplete (and/or non-trusted) information,

– design of interaction and negotiation mechanisms in the situations with changing the synchronous and asynchronous methods of communication,

– methods of construction and exploiting of the acquaintance models and other social knowledge representation mechanisms, and

– use of stand-in agents for representing and acting on behalf of inaccessible actors.

2.2 Suggested Structure of the Project Work

RT1 - Design of a Scenario – based on the currently available scenarios for earthquake modeling, fire fighting and humanitarian coalition operations (in development of some of which the partners were involved in the past) we plan to design and develop a unified reference scenario. The scenario would need to integrate both the model of the environment and the model of the humanitarian/rescue planning and coordination infrastructure.

RT2 - Development of the Multi-Agent Model of the Scenario – based on $\mathcal{A}$**-globe** multi-agent technology (developed at ATG) we intend to develop a software model of the designed scenario. This software system is going to integrate the model of the environment, visualization component and computational models of the rescue actors (e.g. land vehicles, UAVs, field personnel, ...)

RT3 - Development of Activity Oriented Ontologies – deployment of the `<I-N-C-A>` model (developed at AIAI) for representation of distributed, shared and private knowledge required for task planning and activity coordination, as well as refinements of the `<I-N-C-A>` ontology that have to go hand in hand with the distributed planning methods.

RT4 - Planning, Coordination and Replanning of Team-oriented Activities – different methods of distributed planning and coordination (such as peer-to-peer negotiation, acquaintance models, stand-in agents, joint intention theory, shared plans theory but also methods of task distribution, resource allocation and plan merging) will be analyzed. Selected approaches will be deployed within the software prototype developed in RT2. Similarly various approaches to dynamic replanning and reconfiguration will be studied. Examples of team oriented activates may be planning for supply logistics, coordinated flight of UAVs aimed at area surveillance, coordinated movement of robots, land vehicles and UAVs, etc.

RT5 - Demonstration Prototype – In the final phase we will implement several demonstration experiments that will quantitatively analyze the performance of the software prototype featuring the deployed methods investigated in RT4. Besides experimental analysis, the key aim of this research target is to demonstrate the results of the project in an accessible form.

2.3 Phase 1 Preliminary Description of Work

1. investigate and summarize the technical situations and specific tasks for distributed planning in ad-hoc environment
2. preliminary deployment experiments aimed at analysis of how suitable the $\mathcal{A}$**-globe** agent development environment would be for the environment suggested in 1
3. review and analyze the techniques and approaches for the distributed planning that can be potentially used for the tasks listed in 1,
4. elaborate a detailed research plan for the following 15-20 months of the research effort for:
   - *i* design of a formal model of the distributed planning in the ad-hoc environments (contributes to *RT3* and *RT4*),
   - *ii* development or adaptation of the required distributed planning algorithms and (contributes to *RT3* and *RT4*)
   - *iii* development of the experimentation testbed, set of experiments and a demo (contributes to *RT1*, *RT2* and *RT5*).

3 Domain Properties

From the preliminary discussions with the project sponsor, it became evident that the target domain scenario shall include the following properties:

– non-centralized decision making

– limited knowledge sharing

– communication that is unreliable partial inaccessibility, off-line status

– requires dynamic replanning due to strong linkage with the environment

  – reactive replanning
  – predictive replanning

– opportunistic planning

4 Distributed Planning

The problem of distributed planning (DP) has been often discussed in the literature recently⊛ [MP]$_1$:references will be added later. Distributed planning has been viewed as either (*i*) planning for activities and resources allocated among distributed agents, (*ii*) distributed (parallel) computation aimed at plan construction or (*iii*) plan merging activity. The classical work of Durfee divides the planning process into five separate phases, that will guide our further discussion. We intend to comment and update this DP architecture so that it will suit the purpose of the project.

The Durfee DP architecture consist of phases as follows:

$\boxed{1}$ task decomposition

$\boxed{2}$ subtask allocation

$\boxed{3}$ conflict detection

$\boxed{4}$ individual planning

$\boxed{5}$ plan merging

4.1 Task Decomposition

Task decomposition is a classical domain dependent problem. Task decomposition processes work with nontrivial background knowledge or with the data collected or provided by the individual agents. Collecting and providing the data is a typical agent oriented problem that relates closely to all the issues of *data/knowledge sharing*, *data/knowledge disclosure* and *trust*. A related problem would be of task decomposition in the situation where the decomposition knowledge is incomplete. Even though the phase $\boxed{1}$ is to likely to be initiated by a single agent (referred to as DP initiator), we can envisage situations with highly distributed problem solving knowledge where the phase $\boxed{1}$ will be carried in cooperation among several agents.

An easier problem is to find whether there exist a task decomposition, given the various constraints represented by planning knowledge. More complex problem arises in the situations where there are various possible decompositions and the most suitable needs to be selected. In such a cases inter-agent communication is inevitable (this would however overlap with the phase $\boxed{2}$).

4.2 Subtask allocation

Subtask allocation is also a classical problem, which is in distributed environment solved by means of auctions or combinatorial auctions. If the resource availability data are provided centrally the subtask allocation problem reduces to a classical scheduling problem, which can be solved centrally, by existing algorithms. The quality (optimality) of the subtask allocation process depends on the quality of the available knowledge about the other agents (= *social knowledge*).

Allocating the subtasks may fail. In such a circumstances another task decomposition needs to be suggested and processed for subtask allocation. From the conceptual point of view this was meant to be a backtrack situation. However, designers would try to design such a task decomposition mechanism that would comply with feasible subtask allocation. This would be possible if and only if the resource availability data are known centrally.

4.3 Conflict detection

Conflict detection is a phase in which each agent analyzes the requests obtained from the DP initiator. If the request does not match with the agents capabilities or available resources, the agent rejects the respective request. Such a situation is likely to be caused by imprecise social knowledge used during the task decomposition and subtask allocation phases. Another reason for a possible conflict may be the fact that the social knowledge changes since the $\boxed{1}$ and $\boxed{2}$ phases (this can be the case of very dynamic domains).

We claim that splitting the three above listed phases ($\boxed{1}$, $\boxed{2}$ and $\boxed{3}$) of DP on the abstract level, will not result in development of three loosely coupled computational processes. Even though that such a split makes sense from the conceptual view it will not be realistically implemented due to the following reason. The agent who is initiating the task planning process either:

– maintains **high quality social knowledge**, providing very precise information about available resources; in such situations the phases $\boxed{1}$ and $\boxed{2}$ will be be implemented by a single algorithm, or

– there is **little social knowledge** available which results in the phase $\boxed{2}$ being implemented by means of negotiation; if this true then splitting the phase $\boxed{2}$ and $\boxed{3}$ does not make a lot of sense and they will be implemented by a single algorithm.

4.4 Individual planning

Individual planning is a classical plan construction or plan selection activity for which existing planning approaches will be used. If a possible conflict has not been detected during the phase $\boxed{3}$, failure of individual planning is less likely (while can happen in nontrivial planning problems).

More specific reason for failing individual planning (and thus causing a backtrack in the Durfee's planning sequence) occur in the situations that involve *nested planning*. In such situations when there is one specific task, in an hierarchy of the tasks, that looks individual from the DP initiator's point of view, while it requires further decomposition and subtask allocation when teating by the responsible agent.

4.5 Plan merging

Plan merging is a very challenging phase within the Durfee's architecture. However, we argue that in the DP problems there are little practical requirements for obtaining and maintaining a centralized plan within the knowledge structures of the DP initiator. That is why classical works on plan merging are dispensable in the DP context. Instead, we suggest the core of the phase $\boxed{5}$ to be in plan coordination (refereed to as Multi-agent Plan Coordination Problem – MPCP), so that resources are used exclusively used and partial goals shared appropriately.

4.6 Replanning

Replanning is an obvious outcome from a possible failure of the phase $\boxed{5}$ operation. In the ideal case only the individual plans get replanned, while backtracking to phase $\boxed{1}$, $\boxed{2}$ and $\boxed{3}$ is possible. Replanning may also occur (and is very likely to) during the **plan execution** phase or in the idle times, prior plan execution starts. The key triggers of such replanning are (*i*) plan miscoordination during $\boxed{5}$ or (*ii*) deviation from the agents' individual plans. Consequently, one of the key requirements for the DP architecture is a mechanism for **committing** the agents to the individual plan and mechanism supporting coordinated release of the commitment. Also it is expected that planning knowledge availability during the time of planning and replanning may vary or may even become inaccessible.

5 Techniques Supporting Distributed Planning

The good choice of plan representation is important. We need to decide what degree of flexibility would be required for execution. Shall we allow nested planning with the same community of agents, or will that be treated as a separate planning process (both are non-trivial)?

The choice of the plan representation language shall comply with the requirements for the (*i*) specifics domain we may come-up with (*ii*) underlying reasoning/planning processes and (*iii*) openness to support agent-to-agent negotiation and knowledge sharing. Plan representation is likely to be based on HTN, PGP, GraphPlans or POGP. Plan representation and the choice of an appropriate planner supports mainly the phase $\boxed{4}$.

In the following text we present several hand-picked techniques and approaches that may provide a substantial value to the DP integrated architecture.

5.1 Iterative Query-based Acquaintance Model

IQBM is a special contracting protocol that facilitates efficient task decomposition and subtask allocation processes in the situations where little or no social knowledge is available (latter option from above). IQMB is an iterative process of running the phase $\boxed{1}$ based on rather imprecise social knowledge and than merged phases $\boxed{2}$ and $\boxed{3}$. A possible failure of $\boxed{3}$ provides $\boxed{1}$ with additional social knowledge that is used for a refined $\boxed{1}$ process.

- $\oplus$ very flexible and computationally efficient approach, works nicely in semi-trusted environment
- $\ominus$ known weaknesses in competitive environment, tested on very limited datasets

5.2 ECNP/PAP

ECNP – Extended Contract Net Protocol and PAP – provisional Agreement Protocol are specific approach to solving the combinatorial auction problems. As well as IQBM, they also contribute to solving the $\boxed{1}$, $\boxed{2}$ and $\boxed{3}$ DP phases. ECNP extends the protocol with a `provisional accepts` and `provisional rejects` and allows CNP backtracking. Planning here is searching through a dynamically constructed AND/OR graph. PAP also allows `provisional bid` and `withdraw bid`.

- $\oplus$ It has several applications in the military logistics, it has been connected with trust modeling
- $\ominus$ does not work in semi-trusted environment

5.3 Multiagent Opportunistic Planning

A very specific technique for collaborative planning and collaborative plan execution that is making the best use of sharing resources and sharing overlapping goals. In the DP architecture the MAOP contributes to phases $\boxed{2}$ $\boxed{4}$ and $\boxed{5}$.

The key idea is that each agent creates plans that also include opportunities for the other agents. If the opportunity goal becomes pending it can be achieved by other agents. Goals may became unachievable due to changes in the environment or were unachieveble from the very start.

They work with POPG as they provide a bigger deal of flexibility for execution. They have tested several different strategies for selecting the additional goals for which an opportunity may arise.

- $\oplus$ based on minimal knowledge sharing (they share information about other agents capabilities and assigned goals, which may be even too much in our domain)
- $\ominus$ current implementation does not allow for online replanning (in a sense of dropping plans and adopting new plans instead)

5.4 Commitments/Decommitments

There is a broad area specifying formal models of agents commitments as mental structures in their programmes. An inseparable part of each commitment is a specification of the conditions/postconditions under which the agents are allowed to drop their commitments.

There is different use of commitments in the collaborative and competitive environments. While in cooperative settings the commitments postconditions provide mainly notification functionality, in the competitive environment the commitment postcondition provide incentive for an agent to keep its commitment (mainly in the from of penalties). It is believed that the combination of both would be necessary for DP architecture.

This work supports to the phases $\boxed{3}$, $\boxed{4}$ and $\boxed{5}$ of DP architecture.

- $\oplus$ well founded theory
- $\ominus$ not used for an application yet

5.5 Multiagent Plan Coordination

That is a rather theoretical work that is working with partial order, causal link (POCL) definition of a plan. They provide formal definition of the multi-agent parallel POCL plan, where they introduce *parallel step thread flaw* and *plan merge step flaw*. Based on this they have designed a multi-agent plan coordination algorithm that is working with the space of complete plans of the individual agents. The algorithm is based on branch-and-bounds search.

This work clearly supports to the phase $\boxed{5}$ of DP architecture.

- $\oplus$ high relevance, good formal foundation, provides empirical comparison to classical work of Yang
- $\ominus$ centralized, not used for an application yet

5.6 Stand-in Agents

A very specific multiagent technique supporting interaction among the agents while inaccessible or off-line (due to intentional logging off from the network or due to inaccessibility caused by properties of e.g. an ad-hoc networking environment). Stand-in agent is a copied agent that either becomes on-line when the owner is off-line or migrates to such a part of the network that retains its connectivity with the other agents.

– Various distributed methods for optimal placement of the stand-in agents have been designed and investigated (such as *forward swarming* and *backward swarming*).

Stand-in agents are expected to support the DP architecture in the phases $\boxed{1}$, $\boxed{2}$, $\boxed{3}$ and $\boxed{5}$.

$\oplus$ implemented and tested in ad-hoc networking environment

$\ominus$ integration with DP architecture may not be straightforward

6 ATG contribution

– $\mathcal{A}$-**globe** multi-agent environment provisioning and adaptation, collaboration on the computational model of the scenario

– deployment and adaptation of IQBM, ECNP, Stand-in agents

– work on the formal model of DP architecture and integration of further methods

– deployment of methods of computational reflection in multi-agent system (??)

7 Task for the following period

1. design the specification of the domain scenario
2. analyze how existing software infrastructure can support this scenario
3. carry out some preliminary integration exercise
4. plan for a final demo
5. collect the appropriate DP techniques, suggest an integrated DP architecture
6. write a report (and a publication)
7. prepare a follow-up proposal