České vysoké učení technické v Praze
Fakulta elektrotechnická

Czech Technical University in Prague
Faculty of Electrical Engineering

Doc. Dr. Ing. **Michal Pěchouček**, M.Sc.

Plánování autonomní akce v multi-agentním prostředí

Planning autonomous actions in multi-agent environment

Summary:
Current methods of automated planning and multi-agent coordination, represent (*i*) level of sophistication of artificial intelligence embedded in computational systems and (*ii*) an important set of techniques that support numerous applications requiring non-trivial planning and coordination of autonomous robotic and intelligent software systems. Both research fields build on top of and extend the logical foundations of computer science, results from the game theory, methods of distributed optimization to name few. There is a strong potential in advancement of the current state-of-the-art in both fields in coordination, so that methods and approaches can be designed for (*i*) planning in semi-trusted distributed environment, (*ii*) planning in environment with adversarial behavior and (*iii*) real-time planning dynamically changing environment. As there is an application potential in deployment areas such as civilian air-traffic control, coordination of multiple unmanned aerial vehicles, provisioning of ad-hoc networking or persistent surveillance in a dynamic environment, the grand challenge of either research field is to reach out to early adopters, and communicate the research results by means of scalable software prototypes that can illustrate quality, efficiency and robustness of the research results with respect to the applications. This paper provides an introduction to the field of multi-agent planning, analysis of available techniques and an original abstract, component-based multi-agent planning architecture in a response to existing distributed planning architecture [9]. Properties of the designed architecture are discussed and illustrated on three selected applications: air traffic control, production planning and supply chain management.

Souhrn:
Metody automatizovaného plánování a multi-agentní koordinace představují na jedné straně (*i*) vysokou míru sofistikovanosti algoritmů umělé inteligence integrovaných do různých výpočetních systémů ale i (*ii*) důležitou sadu technik, které podporují různé aplikace vyžadující netriviální plánování a koordinaci autonomních robotických a inteligentních softwarových systémů. Obě výzkumné disciplíny stavějí na logických základech počítačových věd (informatiky), rozvíjejí výsledky z teorie her, metod distribuované optimalizace a mnoha dalších. Existuje významný potenciál pro rozvoj stavu poznání v obou disciplínách koordinovaně, tak aby metody a přístupy mohly být použity pro návrh plánovacích systémů v distribuovaných systémech s omezenou mírou vzájemné důvěry, v silně kompetitivních prostředích a pro plánování v reálném čase, použitelné pro rychle se měnící prostředí. Díky tomu, že existuje významný aplikační potenciál v doménách jako je civilní řízení letového provozu, koordinace bezpilotních leteckých prostředků, udržování komunikačních ad-hoc sítí, nebo nepřetržitý monitoring dynamicky se měnícího fyzického prostoru, považujeme za největší výzvu v daném oboru propojení vědců základního výzkumu s pilotními uživateli technologie pomocí škálovatelných softwarových prototypů, které mohou dokumentovat kvalitu, efektivitu and robustnost výsledků základního výzkumu s ohledem na aplikace. Práce obsahuje úvod do oblasti multi-agentního plánování, rozbor metod a návrh původní, abstraktní, komponentově-orientované architektury multi-agentního plánování jako reakci na uznávanou architekturu distribuovaného plánování [9]. Vlastnosti navržené architektury jsou dokumentovány na třech vybraných aplikacích: řízení letového provozu, plánování výroby a řízení odběratelsko-dodavatelských vztahů.

Klíčová slova:
Multi-agentní systémy, plánování, umělá inteligence
Keywords:
Multi-agent system, planning, artificial intelligence

Contents

1. Introduction

The research field of autonomous agents and multi-agent systems (also referred to as agent-based computing, sometimes less precisely as distributed artificial intelligence), a sub-field of computer science and artificial intelligence, investigates the concepts of autonomous decision making, communication and coordination, distributed planning and learning but also game-theoretic aspects of competitive behavior or logical formalization of higher level knowledge structures, which represent interaction attitude of actors in multi-actor environment. Multi-agent system is a decentralized computational system, often distributed (or at least open to distribution across multiple hardware platforms), whose behavior is defined and implemented by means of a complex, peer-to-peer interaction among autonomous agents, rational and deliberative computational units. The autonomous agent is a special kind of an intelligent software program, that is capable of highly autonomous rational action, aimed at achieving its private objective. Agent can exists on its own but often is a component in a multi-agent system. The agent autonomously decide about its own status and its behavior. The change of the status or the behavior can be requested but not executed from outside of the agent. The agent is capable of reactive, real-time response to changes in the environment and to the incoming requests. At the same time the agent is intentional and maintains its own agenda. An important capability is agent social reasoning, which is a capability to reason about other agents in the community and engage itself in collaborative action. Agent technology provides a set of tools, algorithms and methodologies for development of distributed, asynchronous intelligent software applications that leverage the above listed theories.

1.1. Challenges in Agent-based Computing

The future trends and current research challenges of agent-based computing are many. The researchers continue studying approaches and methods of nontrivial deliberative reasoning in multi-actor, competitive and resource bounded environment.

One of the hot challenges is the concept of trust and adjustable privacy management in open environment[1]. There is a need to design methods that would support the actors in sharing knowledge, data, experience in the environment with multiple actors trusting each other differently – in particular the knowledge representation methods and reasoning algorithms supporting the actors to analyze the trade-off between how much of new information can be acquired by knowledge sharing and how much of private information would be disclosed. Related research challenge investigated currently by the research community is the concept of trust, trust-related knowledge representation methods and algorithms that model/compute the individual trust (e.g. based on the communication or observed past behavior).

[1]By *open* we mean the environments with a priori (during the design time) unknown set of actors,

The concept of multi-agent systems is very relevant for the field of collective robotics, coordinated interactions and joint actions of various teams of robots. As robotics is truly a real-time domain, another grand challenge of agent-based computing is to study the concepts of resource[2]-bounded reasoning and interaction. The capability to asses the quality of a decision, a coordinated action or a plan in different times of computation/interaction process would support efficient, robust and safe robotics operations. An example of such robotics operation is autonomous aerial vehicles deconfliction where different deconfliction methods need to be deployed based on actually available time-to-collision.

The internet is a typical, large scale, dynamic multi-actor environment. The internet, as a complex interaction ecosystem, also integrates various models of malicious and adversarial behavior. An important research challenge of agent-based computing is to extend the existing strategic algorithms from the cooperative environments towards the adversarial environments. Similarly, the existing algorithms for detecting malicious behavior (such as anomaly detection in network security) shall be complemented by game-theoretic strategic reasoning capability allowing to reason about the potential activity of the adversary, while monitoring his/her behavior.

Important challenge from the AI perspective is the integration of the concepts of multi-agent coordination and negotiation with complex, *heavy-duty* AI reasoning such as theorem proving, model checking, state-space search or planning. While decomposition of such *heavy-duty* reasoning among multiple reasoners has been subject of research already, less progress has been made in coordination among autonomous reasoners by means of sharing complex information about the progress of their reasoning processes. An instance of such a problem is multi-agent planning of collision-free air traffic, when individual planners representing individual aircrafts share partial plans and negotiate about deconfliction maneuvers.

Agent-based computing is an established field of basic research with solid results and strong application potential. Even though there are important applications that have been successfully deployed in industry, there is still a gap between the researchers and early adopters. The early adopters do not communicate their specific problem sets to the researchers and similarly, the researchers are unable to convince the early adopter about added value of their basic research contributions with respect to the industrial requirements. One of the possible ways of bridging this gap is development of scalable multi-agent models of complex distributed systems that can be used for validation and demonstration of the basic research achievements. Important challenge in the field of agent based computing is systematization, component sharing and standardization of development of such models and demonstrators.

[2]Mainly computational time.

1.2. In this paper

In this paper, we have the ambition to generalize our research experience in basic research in multi-agent planning but also in building planning oriented multi-agent systems prototypes and applications. This paper lacks formal technical details, but includes direct references to the work performed by the authors or other researchers. In this paper we challenge the existing distributed planning architecture provided by Ed Durfee and proposed another, less tightly coupled, component based approach. The arguments are supported by three different case studies of implemented multi-agent prototypes. While the technical content (Sections 2, 1 and 4) of the paper is original, the description of the systems and prototypes (Section 3) have been already published.

2. Multi-agent planning

Planning is a specific instance of computation that performs reasoning about actions in term of their *preconditions*, *effects* and *duration*. Often, reasoning about resources, their allocations and schedules is also falsely referred to as planning, while it shall be more appropriately denoted as resource allocation or scheduling. The problem of multi-agent planning as a planning activity distributed among a group of agents has been often discussed in the AI planning and multi-agent research communities recently (e.g. [9], [7], [12], [5]). Different planning problems are currently termed as instances of multi-agent planning, namely:

– centralized planning for activities and resources allocated among the community of distributed autonomous agents resulting in distributed plan
– planning distributed among several autonomous computational entities aiming at centralized plan construction (or distributed plan construction in the sense of above) or
– combination of both problems listed above, where agents engage themselves individually in planing and coordinating their activities, resource sharing and goal completion.

The computation involved in the first case is an instance of classical centralized planner, where the biggest challenge is a sophisticated integration of the planning and resource allocation processes. The second problem set belongs mainly to the class of problems of parallel computation and distributed problem solving. The subject of our research and the matter discussed in this paper is the latter. In our judgement, the challenges of multi-agent planning are in coordination of planning processes performed by a collective of autonomous agents. The agents either produce and execute plans targeting their individual objectives, or they negotiate about each agents' contribution to the global, shared goal. Besides the goal sharing an important problem in multi-agent planning is also resource sharing, the coordination of which must be also negotiated.

2.1. Properties of multi-agent planning scenarios

The problem of multi-agent planning is relevant in specific scenarios and environments that have some (not necessarily all) of the following properties:

– **Non-centralized communities and multi-party involvement** – an environment hosting agent communities with flat organizational structure, with minimal centralized coordination and planning, having decentralized planning knowledge, information about the skills of actors, resource availability, knowledge and goal perception. At the same time the resulting plans cannot be implemented in isolation by a single actor. Coordination and sharing of resources is required and the goals are set and the planning processes can be initiated by several actors simultaneously.
– **Real-time planning and robustness against changes** – a very dynamic environment where both resource availability and goals are expected to be changing during the planning and execution phases. Such scenarios require computationally efficient planning methods and plan representation models that can be executed in the environment that is dynamically changing.
– **Semi-trusted and adversarial environment** – an environment with partial knowledge sharing within the group, where the actors are motivated to keep a substantial part of their private planning knowledge and resource availability information undisclosed. Environment containing non-cooperative actors, whose goals and motivations go against each other. Environments containing adversaries, intruders and deceptive agents.
– **Varying interaction availability** – an environment based on a communication infrastructure featuring partial and temporal inaccessibility due to e.g. ad-hoc networking, unreliability of the communication infrastructure or actors' frequent change of their off-line/on-line status

Such a set of properties of requirements is typical for rescue operations, complex humanitarian missions, multi-national coalition operations as well as small size military combat operations. These properties are also very typical for the mobile robotics problems, UAV free-flight and MANET (Mobile Ad-Hoc) networking scenarios. Some of these features are also typical for a completely different set of application domains such as virtual organizations, social networking but also production planning. Adversarial multi-agent planning methods are applicable in e.g. wargaming scenarios, network intrusion protection applications and various security and surveillance applications. A typical adversarial planning benchmark is the robocup soccer competition containing several leagues for testing various robotic interactions and coordinated behavior in the environment with a opponent.

2.2. Component-based multi-agent planning architecture

There exists four classes algorithms that are indispensable for design and development of a multi-agent planing system: (*i*) centralized methods of *automated planning*, (*ii*) methods of *negotiation* and combinatorial auctions, (*iii*) methods for representation and maintenance of social knowledge denoted as *acquaintance*

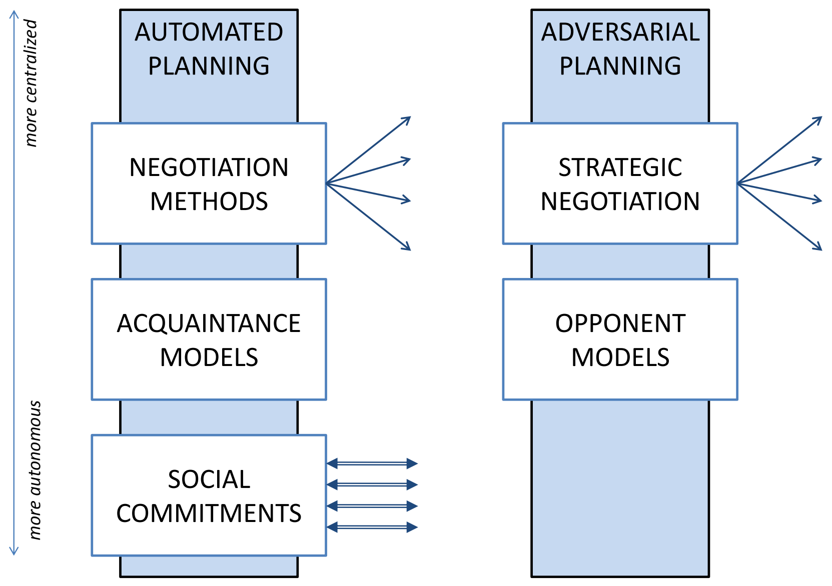

Figure 1. Multi-agent planning components in cooperative (*left*) and adversarial (*right*) environment.

models and lastly (*iv*) *social commitments* methods. Linkage among these methods is loose and depends on a particular application or a system (see Figure 2.2 *left*). The automated planning methods (that can be complemented with the resource allocation methods) still play a central role. Unlike in classical planning, here the planning methods need to be able to work with the negotiation methods in order to learn about skills, availabilities, costs, etc. of the cooperating agents. As communication explosion in practical applications would be far too large, each planning algorithm caches in the acquaintance models the information about the others based on past interaction patterns and observations. Result of the planning process are not actions but commitments of the individual agents towards the joint plan. These are represented by means of the social commitments, well researched knowledge representation mechanisms.

In different environment and scenarios (according to Section 2.1) these components would have different importance and different mode of operation. In the non-cooperative and adversarial environments (see Figure 2.2 *right*), this architecture needs to be slightly modified and enhanced. The automated planning methods will need to be replaced with the adversarial planning methods, which instead of the search through the space of actions perform a search through the game tree (explained in Section 2.2.5). The negotiation capability will need to include the methods of strategic, non-cooperative negotiation and the methods for acquaintance model construction will be based on deployment of the opponent modeling methods as discussed in Section 2.2.3.

The combination of the cooperative and non-cooperative multi-agent planning will be suitable in domains, containing a group of cooperative agents wrestling against another group of cooperative agents (such as robotic soccer).

2.2.1. Automated planning. Methods of automated planning process (i) the information about the *initial state* of the agent, (ii) the *goal predicate* that ought to be valid in the *goal state* and (iii) a set of state transition operators, that represent the actions. The planners provide as a result of their computation a plan, i.e. partially (some times even totally) ordered set of operators, that, if executed, guarantee transformation of the world from the initial to one of the goal states.

As automated planning is an important and traditional branch of artificial intelligence, it produced a remarkable set of various planners suitable for different planning tasks. The classical approach to planning is based on sophisticated search methods searching through the state-space of the operators. Examples of such are linear or nonlinear planners are e.g. STRIPS [17], POPLAN [29] or NONLIN [45].

Lately, there have been established new approaches to planning based on representation of the planing problem by means of an oriented leveled graph – *Planning graph.* The planning graph represents the state space of applicability of the planning operators and various constraints embedded in the planning problem. The planning problem is transformed into a much optimized graph analysis process. As the graph-based planning has been a popular research area, there has been a wide range of available planners that exploit this idea. GRAPHPLAN, the first planner based on the planning graphs has been developed by Arvim Brum and Merrick Furst[2]. DPPLAN is a similar general-purpose planner for STRIPS-like domains. The search algorithm is based on a variation of Davis-Putnam procedure (used in the propositional logic) applied on the graphs complemented with several strategies of choosing the expansion node [1]. LPG is a planner that is based on local beam search (a particular AI technique), and has been designed to handle PDDL (Planning Domain Definition Language) [16]. GRAPHPLAN has been extended by its capability to handle heuristics based on the length of the relaxed plan in the STAN planner [4].

Besides graph-based representation, the planning problem is often represented by means of Binary decision diagram (BDD), a data structure that is used to represent a Boolean function. The BDD planners assume that the planning problem is represented as an entailment problem in fluent calculus (approach to 'change' representation in the first order logical calculus, a variation of the situation calculus). This representation is then mapped onto propositional logic so that the plan, if it exists, can be reconstructed from models of the propositional formulae. Thus for planning, the various available model-checking algorithms (available in the AI community) can be used. BDD planners find the shortest plan. If there is no plan for a particular planning problem, BDD planners can prove non-existence of the plan. BDD planners are able to reuse their results and generate many solutions for the planning problem, or a subset of its results for the planning problems with different initial states. Unfortunately, the BDD diagrams tend to be very complex and thus the process of model-checking, hence planning, is computationally demanding. BDD planners carry out a symbolic breadth first search in the reachability tree of the planning problem. There are number of BDD planners available in the research community. BDDPlan was developed by Hans-Peter Stoerr at the Dresden University of Technology [43]. MIPS (Model Checking Integrated Planning System) [10], which is an alternative planner has been developed at the University of Freiburg, handles STRIPS relations and allows negative preconditions and universally quantified conditional effects. BDD planning can be also implemented by a general OBS, an OBDD (ordered BDD) search engine implemented at CMU [20]. ProPlan (Propositional Planning) is a BDD planner based on simple breadth-first search developed by Michael Fourman, University of Edinburgh [14].

An alternative to model-checking as a reasoning mechanism running in BDD planners is the use of SAT solvers. SAT solvers (e.g. WALKSAT [38] or SATZ [24]) are search-based algorithms that are solving the problem of satisfiability of Boolean formulas. SAT-based planners encode the planning problem into Boolean formulas and solve the satisfiability problem by a selection of SAT solvers. Example of a SAT-based planner is BLACKBOX planner, developed at the University of Washington [21].

2.2.2. Negotiation. The capability to negotiate is absolutely critical for implementation of any multi-agent planning algorithm in the sense as defined in Section 2. Negotiation is a capability that allows reaching a mutual understanding on a specific matter among the group of agents. The specific subject of negotiation can be task allocation, profit division, shared resources deconfliction or teamwork coordination. The group of agents can be totally peer-to-peer or it can be hierarchical. An important property of negotiation is that each agent is fully autonomous and accepts the result of negotiation if it provides the agent with a profit (increased individual utility).

The negotiation mechanism is defined by:

1. *Negotiation set* – a set of possible outcomes from negotiation, where the negotiation process selects one outcome of the negotiation set
2. *Negotiation protocol* – a protocol that specifies in which order do the agents present each other with the bids, what are the required properties of the bid and what are the properties of a possible counterproposal, should the original bid be rejected,
3. *Deal/stuck identification rule* – specification of the circumstances under which the bid is accepted by the both agents and can be declared as a *deal* or when it becomes clear that the deal cannot be achieved and a *conflict* is declared.

Behavior of each agent in the negotiation is defined by a *strategy*, which selects the particular bid out of the set of possible bids defined by the 1 and the 2 above. Very often the strategies are private. There are different game-theoretical techniques that analyze mutual relation of the agents strategies with respect to their Pareto-efficiency and stability (e.g. Nash equilibrium). Agents may negotiate about multiple issues at the same time.

Negotiation can be divided into three different classes:

– *1-to-1 negotiation.* Here two peer agents are trying to reach mutual agreement on specific matter matter. The most of the 1-to-1 negotiation techniques are based on variations of the *monotonic concession protocol (MCP)* negotiation protocol [11]. In MCP the agents take turn in proposing each other the bid, until the utility of the first agent's bid from the perspective of the second agent provides better profit than utility of his own bid. If this condition is not met, the agents may provide each other counterproposals, where each of them needs to be less profitable than the original bid ($u_B(\delta_A) < u_B(\delta'_A)$). There exits a specific strategy, known as Zeuthen strategy that specifies under which conditions it is better for the agent to concede and when he better declare a conflict deal [18]. The Zeuthen strategy suggest that the agent shall concede if his opponent's willingness to risk a conflict is greater than willingness of that agent. The willingness to risk is a value from $\langle 0, 1\rangle$ and is defined as follows:

$$risk_A^t = \begin{cases} 1 & \text{if } u_A(\delta_A^t) = 0 \\ \frac{u_A(\delta_A^t) - u_A(\delta_B^t)}{u_A(\delta_A^t)} & \text{otherwise} \end{cases}$$

  Provided that the deal δ_A^t provides the agent A with no profit, he better declares a conflict as he cannot gain anything from the deal. Otherwise the risk is based on the differences between the utilities of the two specific deals – the closer they are, the less likely is that the agents would declare the conflict. The result that MCP provides has got an important property. It maximizes the product of utilities of the two agents, hence the result is Pareto optimal[3]. Consequently the the agents are motivated not to deviate from the deal.

– *1-to-many negotiation.* Here, one agent is trying to reach an agreement with a collective of agents. Typically, the agent denoted as a *requestor* is trying to select the agent denoted as a *provider*, who provides the requester with such a bid (for the service) so that marginal costs for subcontracting the service to this particular agent is the least. The 1-to-many negotiation protocols are often based on the variation of the *contract-net-protocol* (CNP) [42], which is computational variation of the *seal-bid auction.* While for the individual service this approach is optimal, for the sequences or batches of tasks CNP is suboptimal. In order to tackle this problem, CNP has been improved recently

[3]There is no other deal that can provide the one agent with greater utility, while not lowering the utility of the other.

by integrating the backtracking capability in *extended-contract-net-protocol* (ECNP) [13] or *provisional-agreement-protocol* (PAP) [34]. Combination of bidding for individual bids, clusters of bids, and mere swapping of the bids have been theoretically studied in [37]. In different settings, where the agents strategies are private or semi-private, different protocols are recommended. For example in the *Vickery auction*, the seal-bid second-price auction is beneficial for the requestor as the rational agents are motivated to bid their true valuations. Often used are also the *English auction*, the *Dutch auction* but also the *All-pay-auction*, which is very conservative as it requires the agents to pay each time they propose the bid (similar to the game of poker).

– *Many-to-many negotiations.* These negotiation protocols have the objective to achieve a shared opinion, a shared goal or a mutually accepted task allocation among a number of peer agents. An obvious approach would be based on an aggregated 1-to-1 negotiation methods (e.g. as discussed in the Section 3.1) or multiple parallel runs of 1-to-many negotiation methods. The latter has been widely used e.g. in the telecommunication industry for load balancing of the network of signal providers (here each agent can be provider and requestor in the sense of CNP at the same time). Besides these methods the field of agent-based computing is using the various existing voting protocols. The product of the voting protocols is a *social choice function* that represents in the best possible ways the preferences of the individual agents across the possible outcomes of the voting. The resulting social choice function shall (*i*) exits for all possible inputs (individual preferences), (*ii*) be defined for every pair of outcomes, (*iii*) be asymmetric and transitive, (*iv*) be Pareto-Efficient with respect to all agents' preferences, (*v*) shall be independent of irrelevant alternatives (looser cannot turn the winner) and (*vi*) no agent shall be a dictator. There exists the famous Arrows Impossibility Theorem [15] that claims that no social choice rule satisfies all of these six conditions. The most used voting protocol is the *plurality protocol*, where all alternatives are compared simultaneously. Here introducing an irrelevant alternative splits the majority, which dissatisfy the requirement (*v*) from above. Another alternative is the *binary protocol* where all alternatives are voted pair-wise and the winner challenges further alternatives. While here the looser cannot turn the winner, the downside of this method is that its result depends on the initial pairing and different results can be provided by different pairing. Quite popular voting protocol is the *Borda protocol*, that assigns values to outcomes according to how high are they in someone's preferences lists. Similar to the plurality protocol in the Borda the irrelevant alterative may turn the winner.

2.2.3. Acquaintance models. While the previous two components of multi-agent planning were rather general now we will discuss two very specific research methods that play an important role in the multi-agent planning. The acquaintance model represents the agents mutual awareness and collects the information the agents know one about the other - *social knowledge* [28]. Some of the social knowledge can be shared intentionally, some not. The agent's social knowledge can be made available from previous interaction or provided by an independent monitoring mechanisms. We distinguish among different classes of social knowledge:

- *Minimal social knowledge* – collection of information about all agents that the respective agent is aware of (the information about their IP physical addresses, port number, their ACL language they use for communication). The minimal social knowledge represents minimal and mandatory requirements for interaction among the agents (provided by e.g. AMS in Jade).
- *First level social knowledge* – collection of information about the services and skills that the particular agent provides to the community. This kind of social knowledge improves the collaborative properties of the multi-agent system. In the non-collaborative environment the first order social knowledge may be regarded as a private information with respect to a part of the community.
- *Higher level social knowledge* – knowledge about agent's outer characteristics such as of agents awareness of agents' reliability, maintain and manipulate trust, each other communication, computational and operational load but also information about price and a completion time. In the adversarial environments this type of knowledge supports agent's private knowledge reconstruction, speculation about intentions, other mental models, models of future agents behavior. In this circumstances it is often referred to as an opponent model.

There are different ways how the acquaintance models can be constructed and maintained. Centralized monitoring is easy to implement in multi-agent systems and it avoids possible duplication and redundancy. However, it may become a bottleneck in large scale or real-time applications, and is absolutely inappropriate in the applications in the semi-trusted and adversarial environments. In many multi-agent systems, the acquaintance models are maintained by the dedicated agents, loosely coupled agents such as middle-agents, brokers, matchmakers, mediators, who provide first-level and higher-level social knowledge and negotiate an effective forms of cooperation. More sophisticated approach to social knowledge maintenance is by means of direct communication and interactions in the multi-agent community. Various 'pull models' (such as *periodical revisions*) or 'push model' (such as *subscribe-inform protocol*) are available. In the adversarial and semi-trusted environments social knowledge is often constructed by means of opponent modeling methods based on independent monitoring of the surrounding agents and using various data-analysis and machine learning methods for generalization of appropriate social knowledge.

There have been several specific architectures of the acquaintance model designed in the past – *tri-base* (3bA) acquaintance model [30], *twin-based model* [3], acquaintance model in ARCHON [44]. All of these were collecting primarily the first level social knowledge and basics of higher level social knowledge. As these models were build in order to provide an alternative to the social maintenance provided by the dedicated agent, they were maintained mainly individually by communication.

We have designed a specific negotiation protocol – Incrementally Refined Acquaintance Model (IRAM), that is used for optimizing the size and the quality of the acquaintance models and thus providing very efficient while privacy preserving negotiation and contracting (see Section 3.3 and [35]).

2.2.4. Social Commitments. The social commitment is a knowledge structure describing an agent's obligation to achieve a specific goal if a specific condition is made valid. It also expresses how the agent can decommit from this obligation. The commitment does not capture description how the committed goal can be achieved. Individual planning for a goal achievement, plan execution and monitoring is a subject of agents internal reasoning processes and it is not represented in the commitment. In the context of the planning problem defined in Section 2, we understand the agent's specific goal (to which it commits) as an individual action, a component of the plan, which resulted from the given planning problem. While typical action in the plan contains only a precondition and an effect, in multi-agent planning this representation can be extended so that the commitment-related information is included. The commitments are vital for multi-agent planning as they increase robustness of the final plan against the changes in the environment.

Michael Wooldridge in [47] defines the commitments formally as follows:

$$\begin{aligned} &(\mathsf{Commit}\ A\ \psi\ \varphi\ \lambda),\\ &\ \ \lambda = \{(\rho_1,\gamma_1),(\rho_2,\gamma_2),\ldots,(\rho_k,\gamma_k)\}, \end{aligned} \tag{1}$$

where A denotes a committing actor, ψ is an activation condition, φ is a commitment goal, and λ is a convention. The convention is a set of tuples (ρ,γ) where ρ is a decommitment condition and γ is an inevitable outcome. The convention describes all possible ways how the commitment can be dropped. Generally speaking, the actor A has to transform the world-state in such a way that the φ goal becomes true if ψ holds and any γ has not been made true yet. The actor is allowed to drop the commitment if and only if $\exists i : \rho_i$ which is valid. A decommitment is allowed provided that γ_i is made true. A formal definition in modal logic (working with the models of mental attitudes like Believes, Desires, Intentions, [40], and temporal logic where the operator AG denotes an the inevitability) as defined in [47] follows:

$$\begin{aligned} (\mathsf{Commit}\ A\ \psi\ \varphi\ \lambda)\ \ \equiv\ & \\ & ((\mathsf{Bel}\ A\ \psi) \Rightarrow \mathsf{AG}((\mathsf{Int}\ A\ \varphi) \\ & \quad \wedge(((\mathsf{Bel}\ A\ \rho_1) \Rightarrow \mathsf{AG}((\mathsf{Int}\ A\ \gamma_1))) \curvearrowleft \gamma_1) \\ & \quad \ldots \\ & \quad \wedge(((\mathsf{Bel}\ A\ \rho_k) \Rightarrow \mathsf{AG}((\mathsf{Int}\ A\ \gamma_k))) \curvearrowleft \gamma_k) \\ &) \curvearrowleft \bigvee_i \gamma_i). \end{aligned} \tag{2}$$

This definition is used in a declarative way. Provided that whatever the agent does during a specific behavior run complies with the above defined commitment, the expression 2 is valid throughout the whole duration of the run.

There are different commitments widely used in the multi-agent community:

– *Minimal social commitment* – contains the least binding type of decommitment rule requiring the agents to notify the members of the team about its inability to achieve the commitment, should it decide to drop the commitment.
– *Relaxation* - is a special decommitment, triggering a new negotiation round aimed at replacing the commitment with a new relaxed commitment (such later due time, increased resources requirements, or lower quality of service).
– *Delegation* - by using this type of commitments the agent shall be able to find some other agent who will be able to complete its commitment on the original agent's behalf. It is possible that such a commitment will contain unbound variables representing the need to search for an agent suitable for delegation.

Formal work linking commitments and planning in a general manner has not been yet elaborated. However, we managed to perform statical measurement of the different decommitment strategies in overstressed scenarios [46].

2.2.5. Adversarial Reasoning and Adversarial Planning. While adversarial reasoning/planning is set slightly separate from the core of the multi-agent planning research, it is a relevant approach satisfying the adversarial property of the target problem set and deployment environment. *Adversarial reasoning* is loosely defined as reasoning about motivations, intentions and behavioral patterns of the adversary, with the objective to perform an action that would maximize own profit or neutralize the adversary. Adversarial reasoning is also known as *opponent modeling*, an activity that construct the opponent model, a knowledge structure containing high-level social knowledge. *Adversarial planning* is understood as planning activity producing plans leading towards the goal with consideration of the interference of actions of the adversaries.

The most of the methods for adversarial planning are based on sophisticated methods of *adversarial tree* or *game tree* search. Formally, the game tree is a perfect- or imperfect-information game representation in the *extensive form*[4]. According to [39], the perfect-information game tree is represented by (*i*) a set of players, (*ii*) set of actions, (*iii*) set of nonterminal choice nodes, (*iv*) set of terminal choice nodes, (*v*) the action function, which assigns to each choice node a set of possible actions, (*vi*) the player function, which assigns to each nonterminal node a player who makes next turn, (*vii*) the successor function, which maps a choice node and an action to a new choice node and (*viii*) finally a real-valued utility function for the player on the terminal nodes.

[4]differs from the *normal form* and various induced normal forms representations in a sense that it only represent the combination of actions of the agents with the associated utility

Provided that the utility function of the opponent is known, assuming opponent's rationality with respect to its utility function and assuming that the agents take turn in performing the actions, the game tree is an appropriate representation of the course of interaction in the game. The *dominant strategy* (such as strategy of the agent, that cannot be outperformed by any other strategy with respect to the utility of the terminal node) can be determined by searching the game tree, by e.g. *Minimax* algorithm [8].

The real adversarial interactions are slightly more complex than suggested in the previous paragraph. First, actors do not need to keep the turns in which they take actions and they can act concurrently (an example of such a game is *asynchronous chess*). Besides, the complex real interactions include multiple players (solved by extending the *minimax* algorithm to *maxn* algortihm [27]). Also only partial knowledge about the non-deterministic outcome of the action function. The research community has made tremendous progress in this direction by providing formalism for modeling the games with incomplete information (e.g. Baeysian games [19]). While in the academic games, symmetry of the utility function is assumed (games denoted as zero-sum games, where the total sum of the agents utilities remain constat throughout the game), in real complex games, the utility is asymmetric (non-zero sum). Also the utility function of the players does not need to be identical, but it is also only partially related.

The critical research challenge for deployment of the adversarial planning in the multi-agent domains such as robocup soccer or intrusion detection is the latency of the response. These domains are typical for voluminous, nontrivial game trees and requirement for near to real-time response. There are various novel game-tree representations that consider the computational challenges of game-tree search algorithms, such as *graphical games* [22], *action graph games* (AGG) [6] or *Multiagent influence diagrams* (MAIDs) [23]. An alternative way how to deal with adversarial planning complexity is to adopt richer opponent models and problem solving heuristics. Provided that there exist an opponent modeling algorithm that identifies the goals of the opponents, an original algorithm *Goal-based Game Tree Search* (GB-GTS) reduces the game-tree substantially and thus allows the algorithm to searcher in a greater depth of the tree [25]. Agent Subset Adversarial Search (ASAS) algorithm reduces the game tree by putting limits on the number of interaction agents during the game [26].

2.3. Abstract Distributed Planning Architecture

The classical work of Durfee [9] provides an abstract architecture for distributed planning that fits in parts the requirements listed in Section 2.1 and will be used as a reference for further discussion. Durfee divides the planning process into five loosely coupled separate phases, namely:

1. *task decomposition*, when a shared goal or user defined task is understood by one or several agents and the decomposition of the shared goal into tasks is provided (usually by a single agent),

2. *subtask delegation*, when the individual tasks are allocated to the agents, usually based on background knowledge about agents skills and capabilities
3. *conflict detection*, when the tasked agents confront the requests with their real skills and capabilities, but importantly with their plans and already made commitments (with the intention to identify conflicts if they exist)
4. *individual planning*, when the individual agents deploy centralized planning algorithms to plan the course of their actions aimed at achieving the requested task (this phase may also result in a conflict should no plan exist) and
5. *plan merging*, coordination of the individual plans of the agents so that the use of resources in the environment (such as space) is non-conflicting.

As mentioned earlier, the abstract distributed planning architecture matches the requirements for multi-agent environments only partially. In this section we will provide a thorough discussion of the extend to which the present architecture is suitable for the defined problem set.

2.3.1. Non-centrality and multi-party involvement: The requirements for *non-centrality and flat hierarchy* of actors and the requirements for *multi-party involvement* are the chief bottlenecks in deployment of the Durfee's distributed planning architecture. Durfee assumes that there is a single actor who is responsible for decomposition phase of one task. In contrary, these requirements assumes not only that there is no agent who can perform the requested task, but mainly that there is no agent who can plan how the requested task shall be performed. Consequently, the task decomposition and subtask delegation phases can be hardly implemented by a centralized algorithms. While task decomposition phase is concerned with availability of the problem solving knowledge for planning, subtask delegation phase needs the right amount of social knowledge that is needed for an appropriate resource allocation.

Non-centrality and multi-party involvement is a vital multi-agent property and is supported by the two multi-agent planning components: negotiation and acquaintance models. In the task decomposition phase these requirements are addressed by the contract net protocol or various auctioning algorithms. Non-centrality is however more critical in the subtask delegation phase. The appropriate choice of the subtask delegation mechanism depends on availability and the quality of social knowledge. Should social knowledge be available in a good quality to the agent, who is charged with the subtask delegation process, the classical centralized automated planning and/or resource allocation methods shall be used. Otherwise the subtask delegation problem is to be solved by negotiation as explained in Section 2.2.2. The request is rejected during the conflict phase provided the it does not match with the agents capabilities, resource availabilities or collaboration preferences. Conflict may arise due to (*i*) usage of imprecise social knowledge (caused e.g. by confidentiality reasons, resources overestimation, etc.) used during task decomposition and subtask delegation phases, (*ii*) agents deliberate overconstraining its responsibility during the subtask decomposition process, or (*iii*) very frequent changes of agents availability (and thus changes of social knowledge) since the task decomposition and subtask delegation phases (in case of very dynamic, real-time domains).

We claim that social knowledge availability affects implementation of the proposed multi-agent planning architecture. Let us investigate the two extreme cases:

- There is high quality of higher level social knowledge available, providing very precise information about available resources. In such situations splitting task decomposition and subtask delegation is inefficient and both processes will be be implemented by a single algorithm.
- Only minimal social knowledge is available which results in the phase of subtask delegation being implemented by means of negotiation. During this process the agents will avoid conflicting deals. In such situations the phase conflict detection will be embedded within the subtask delegation phase.

This argument gives an impression that the Durfee's multi-agent planning architecture will be based on the separate task decomposition, subtask delegation and conflict detection processes in all but the listed two extreme cases. However, if the amount and quality of available social knowledge requires at least small amount of interaction in the phases subtask delegation phases (it is unlikely that any sperate phase conflict detection will be required as the conflicts will be avoided during the phase subtask delegation. On the other hand, if there is no negotiation and interaction required during the subtask delegation phase there is no reason for splitting task decomposition and subtask delegation. Consequently splitting the phases is reasonable only in the case where social knowledge availability is different in various situations and within different teams of agents.

2.3.2. Dynamic Environments: The requirements for planning architecture to support very dynamic environment requires the computational process to be well balanced between the particular phases of the multi-agent planning process. Even though the process of planning in the presented environment is not performed in critical-time and very often there is enough time for the use of some of the classical heavy duty planners, for replanning (which may occur frequently) the minimal amount of computation and communication shall be assured.

Replanning as a typical capability needed in dynamic environments needs to be addressed from three perspectives:

- *replanning prevention* – The use social commitment allows construction of such plans that are resilient to specific types of changes in the environment in such a way that changes of the courses of actions, that respond to the changes in the environment, happens during the execution phase and no replanning is required (such as delegation or relaxation classes of decommitments – see Section 2.2.4).
- *replanning organization* – If replanning needs to be performed, backtracking in the Durfeee architecture suggests that replanning is solved first in the phase of individual planning. If no solution is found, the deployment of the revised subtask delegation process within a limited subgroup of collaborating actors

is recommended. If no other delegation can be found, new decomposition shall be produced. In spite that this approach of localized replanning may produce suboptimal solutions, it has proved to be very efficient in the time critical domains (such as air-traffic control, see Section 3.1.

– *replanning efficiency* – The algorithms deployed in the phase of individual planning need to be highly efficient in order to be used for replanning in time critical domains. Thus such a a subset of automated planning algorithms listed in Section 2.2.1 that complies with this requirements needs to be selected.

In conclusion the Durfee's distributed planning architecture as such does not support the process of monitoring and thus replanning capability in very dynamic environment. From the suggested multi-agent planning components the social commitments algorithms and efficient automated planning algorithms support such environments.

2.3.3. Semi-trusted and partially accessible environment: The requirements for partial knowledge sharing and coping with varying interaction availability are linked in the sense that either requirement makes it possible to share needed knowledge only partially. The planning knowledge needed for task decomposition, the social knowledge in the subtask delegation phase, or even information about activity status needed for the plan merging phase are shared only in parts.

The more trusted and more accessible the community of agents is the more we can use the negotiation methods listed in Section 2.2.2 in combination with the automated planning methods listed in Section 2.2.1. With decreasing trust or connectivity, each of the agents need to build its own acquaintance model (by means of methods listed in Section 2.2.3) and use them for reasoning when the particular instance of knowledge is unavailable. The Durfee multi-agent planning architecture does not support directly these properties of the planning problem, however all the phases but delegation and plan merging are unaffected by interaction unavailability.

2.3.4. Adversarial environment: As noted previously, adversarial planning is not in the center of the multi-agent planning research. As the Durfee abstract architecture has been oriented mainly towards cooperative planning and cooperative execution of the resulting plans, it does not support this requirement. This requirement is fully supported by the methods listed in Section 2.2.5.

Let us distinguish two different noncooperative settings of the communities. If the non-cooperative relationship is established among all pairs of the members of the multi-agent community, the adversarial planning methods need to present in the individual planning phase. Strategic reasoning is also expected in the conflict detection and plan merging phases. If non-cooperative relationship is established only among a subset of all pairs of the agents, meaning that there are cooperative subgroups with the system, adversarial planning needs to affect also task decomposition and delegation phases.

3. Multi-agent applications and prototypes

In the following we reflect on three original industry driven multi-agent planning applications in order to demonstrate and discuss the concept of loosely couples planning components listed in the Section 2 and inappropriateness of the Durfee abstract planning architecture introduced in Section 2.3. The following three sections introduce in turn the three applications while in the Section 4 we provide a brief analysis.

3.1. Agend-based free-flight planning

In cooperation with the US Air Force, the US ARMY and the Federal Aviation Administration (FAAA) the researchers in the Agent Technology Center, Czech Technical University, have developed AgentFly - a scalable, agent based technology for free-flight simulation, planning and collision avoidance. In AgentFly each flying asset is representing as a specific software container hosting multiple intelligent software agents. Each agent either models a specific functionality of a platform hardware, such as sensory capability, dynamic flight control or communication or encapsulates an intelligent decision making technology that supports planning or collision avoidance capability. Such an architecture supports three principal use-cases of the AgentFly system: (*i*) multi-agent modeling and simulation of free flight, (*ii*) control of free-flight unmanned aerial platforms and (*iii*) Alternative approach to planning supporting civilian air traffic control.

AgentFly is designed so that there is no centralized component needed and all the planning and collision avoidance is based on sensory capability of the flying asset and their distributed (peer-to-peer) decision making capability. As such the planning and collision avoidance agents can be directly deployed onto hardware platforms and thus can support real free-flight exercise of the unmanned aerial assets. AgentFly system is based on AGLOBE multi-agent technology which supports seamless migration from the computational simulation towards hardware deployment. Previously, an AGLOBE-based model of ground robotic scenario has been successfully migrated towards RoboCup soccer environment.

The most promising direction of AgentFly exploitation is in the area of air-traffic planning. Currently Federal Aviation Authority (FAA) is interested in testing the AgentFly planning capacity for the civilian air traffic in heavily overloaded traffic across the whole National Air Space (NAS). The whole idea behind multi-agent approach to air traffic planning is based on (i) relaxation of the planning problem and (ii) multi-agent flight simulation. Instead of planning a collision-free operation for a high number of aircrafts, there is a flight-plan constructed for each individual aircraft without consideration of possible collisions. Subsequently, such an operation is simulated in AgentFly environment, possible collisions are detected and solved either by individual re-planning or by means of peer-to-peer negotiations.

3.1.1. Planning in AgentFly. Each agent has its own path planner component which provides the smooth flight plan trajectory respecting all constraints given

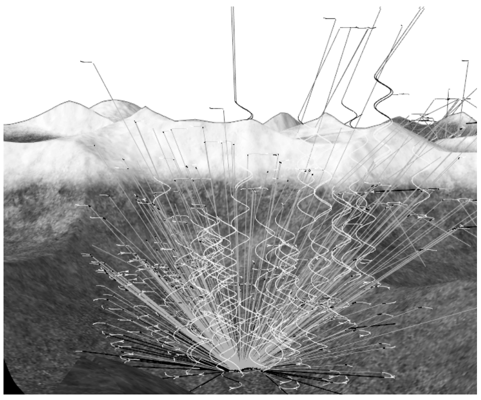

FIGURE 2. Path-planning example in a mountainous environment.

by the airplane model and its goals (mission). Goals for the airplane and requests for collision avoidance maneuvers are transformed to a sequence of way-points where each way-point can has specified time and cruise speed restrictions beside its position. The planner prepares the detailed description of the flight corridor - graphical description and cruise speed changes in time. Internally, the path planning process is running in two coupled phases: (i) spatial and (ii) time planning. During the first phase the spatial part respecting position restrictions of the output flight trajectory is prepared using Accelerated A* (AA*) algorithm [41], see Figure 3.1. The AA* is suitable for fast planning in the large environments where exists many operational restrictions like no-flight zones, minimal flight levels, dynamic zones from non-cooperative collision avoidance as described below. Defined airspace is kept in the tree where each inner node is defined as a composition or transformation operator and leafs keep zone definitions which use the geometrical description, height map and octant tree representations. Using composition of such zones in tree very complex airspaces can be modeled. In the second phase the planner plans the cruise speed changes mapping them to the prepared spatial part of the trajectory. The time planning goal is to fit all given time and cruise speed restrictions. There exists the loop between both phases in order to be able to handle also limit cases, e.g. airplane cannot slow down more below its minimal cruise speed thus the spatial part has to be made longer.

3.1.2. Collision avoidance (CA) in AgentFly. No matter whether used for simulation, planning or real hardware control, the capability of AgentFly to autonomously avoid collisions is in a centre of its research contribution and is critical for its commercial exploitation. AgentFly features three classes of collision avoidance mechanisms:

– **Rule-based CA** (RBCA) algorithm is a domain dependent algorithm based on the Visual Flight Rules defined by FAA. Upon the collision threat detection, the collision type is determined on the basis of the angle between the direction vectors of the concerned aircrafts. Each collision type has a predefined fixed maneuver which is then applied in the re-planning process. Visual flight rule-based changes to flight plans are done by both assets independently because the second asset detects the possible collision with the first asset from its point of view. This mechanism has been implemented in AgentFly for the testing purposes. The added value of the below listed collision avoidance mechanisms was measured against rule-based CA.

– **Iterative peer-to-peer CA** (IPPCA) algorithm deploys multi-agent negotiation aimed at finding the pareto-optimal CA maneuver. Software agents hosted by each asset generate a set of viable CA maneuvers (by means of the planning mechanism described above) and compute costs associated with each maneuver (based on e.g. the total length of the flight plan, time deviations for mission way-points, altitude changes, curvature, flight priority, fuel status, possible damage or type of load). The agents negotiate such a combination of maneuvers that minimizes their joint cost associated with avoiding the collision.
– **Multi-party CA** (MPCA) algorithm extends the above presented CA algorithm by allowing several assets to negotiate about collective CA avoidance maneuvers. While the quality of the overall plan based on the utility-based CA is strongly affected by the order in which the collisions occur and are solved, multi-party CA is has been designed to minimize the effects of CA maneuvers causing conflicts in future trajectories with other flying assets. While requiring substantially more computational and communication resources for solving a single encounter, this strategy has shown to provide more efficient free-flight collision free trajectories in longer runs.
– **Non-cooperative CA** algorithm supports collision avoidance in the case when communication between aircrafts is not possible. Such a situation can arise e.g. when on-board communication devices are temporarily unavailable or when an asset avoids a hostile flying object. This class of algorithms is based on modeling/prediction the future airspace occupancy of the non-cooperative object and representing its possible future positions it in terms of dynamic no-flight zones. Based on this information, the algorithm performs continuous re-planning using the previously described planning algorithm.

Even though that AgentFly can compare effectiveness of various CA methods in different scenarios, the free-flight dynamic environments are rarely suited for use of a single CA algorithm at all times. Therefore AgentFly features efficient multi-layer collision avoidance architecture that provides sophisticated mechanisms for flexible selection of an appropriate CA algorithm in various situations. This architecture features meta-reasoning process that analyze time-to-collision and estimated time requirements for each individual CA avoidance method with respect to efficiency of the collision avoidance process needed. The multi-layer collision avoidance architecture avoidance module works in the fully decentralized manner

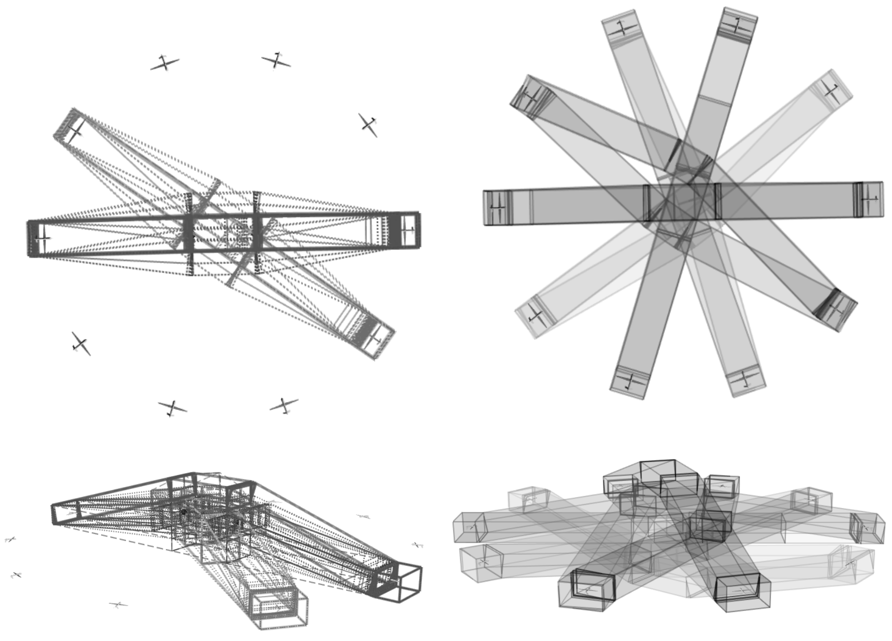

FIGURE 3. Collision avoidance solution after several IPPCA iterations in 3D (b) and in 2D (d) for the superconflict setup of 10 airplanes. Visualization of negotiation 3D (a) and 2D (c).

and does not utilize any central planner for the collision avoidance of physical entities. The architecture is domain independent and therefore is ready for the deployment in the autonomous vehicles like airplanes, robots, cars, submarines, etc. Deployment scenarios and selected experimental results

AgentFly system has been used for validation and testing of decentralized collision avoidance by comparing the selected properties (quality of solution, required computational and communication resources, etc.) of given algorithms. Algorithms are often benchmarked in complicated collision cases, e.g. super conflict scenario where airplanes are located in the circle, all are flying to opposite side of the circle implying the multi-collision of all airplanes in the circle center (in super conflict scenarios (see Figure 3.1.2 as an example)). For comprehensive test results refer to [41] and [33].

3.2. Agent-based production planning

There has been a notable deployment of agent based systems in the manufacturing domain. In several applications the concept of agents has been used for modeling and simulation of the manufacturing process, exceptionally also for real-time production process control. Important manufacturing application area is production planning. This concept has been successfully applied for planning manufacturing processes [32]. The European car company SkodaAuto, a member of the Volkswagen group, requested deployment of multi-agent planning technologies for planning their mass-production of car engines. Technology deployment has been coordinated by Gedas, s.r.o., (T-System software company) who implemented the final product, while the Gerstner Laboratory of the Czech Technical University in Prague (CTU), and CertiCon a.s. have contributed by the design, prototype development, and technical experience.

The factory layout consisted of three closely linked assembly lines (ZK line, RUMPF line, and ZP4 line), two operational storage units (Vehicle store and Conveyor store) and a final products storage capacity. The factory manufactures daily up to 2000 pieces of engine heads for 2 and 4 cylinder 1200 cm3 engines, 2000 pieces of 3 cylinder RUMPF engines, or 1200 pieces of finished 3 and 4 cylinder 1200 cm3 engines (with 66% manufactured for sale). The engines can be assembled either from parts produced in the factory or from parts purchased externally. The functionality requirements for the final planning system were to provide a detailed production plans for a six weeks period, so that storage requirements and consequently also storage cost throughout the production chain were minimized, production type uniformity was maximized, tooling changes were minimized, and any unnecessary handling of products between successive steps of the production process was be minimized. The system shall be also open to integration with production monitoring and management tools and allow further system reconfiguration according to changes in the production processes themselves and support real-time re-planning in the case of demand changes or production anomalies.

Due to very high complexity of the planning problem and several nonlinear constrains a classical operational research methods cannot be used. Instead, the planning problem has been decomposed into high-level planning and low-level planning processes. The former represents the solution to a substantially relaxed planning problem based linear programming. A coarse, 6-weeks semi-optimal production plan is provided as a result of the high-level planning. This plan however does not comply with the various non-linear local production constrains such as knowledge of late arrival of material, rump-up and completion phase of a daily production (respecting the different numbers of shifts), change of tooling, and many others.

During the low-level planning the production process is simulated by a multi-agent system to detect conflicts and inconsistencies in the high-level plan (see the diagram in Figure 3.2). The planning agent sends the high-level production plans to the agents representing the physical entities on the shop floor. The agents use their local resource allocation mechanisms to assign appropriate processes and continually consult dependencies among them. Detected inconsistencies trigger local re-planning algorithms. All agents receive their goals prescribed by the high-level plan. The difference between the necessary and the nominal load defines replanning priority. The agent with the highest priority performs replanning (by means

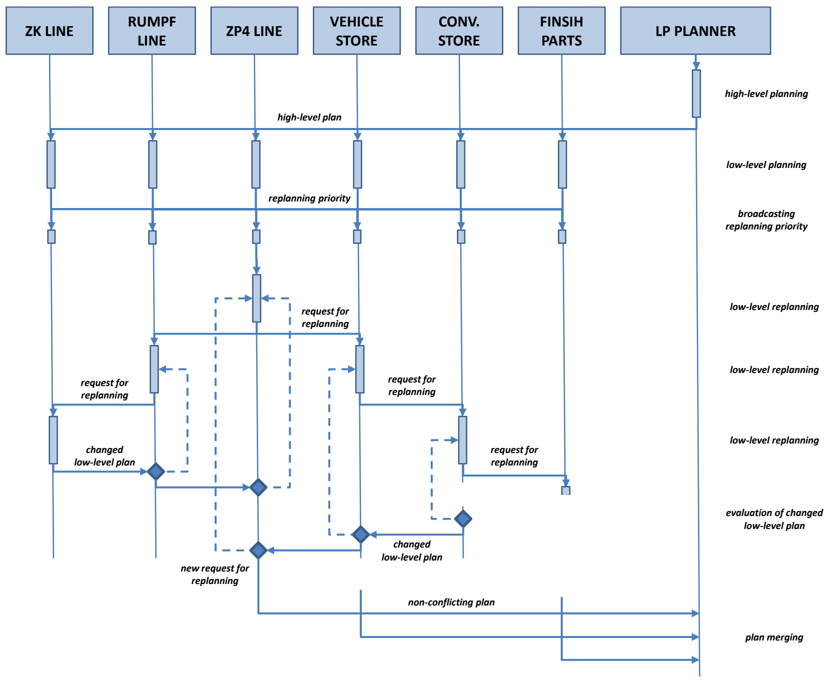

FIGURE 4. Integration of high-level and low-level planning and replanning by means of agent interaction.

of classical search methods) so that all constrains are satisfied, and requests appropriate changes from the agents whose plans are linked by resources. These agents perform replanning in the same manner. The revised plans are sent back to the requestor. This process can be iterated until the changes in the plans of the individual agents comply with all the predefined non-linear constrains. Even though this negotiation process has not been theoretically proved for cycles' avoidance, practical experiments have validated its operation.

The described production planning system in SkodaAuto is an important part of a modular Manufacturing Execution System (MES), which is designed to cover in successive steps all of the eleven areas of functional model of MES. Besides necessary interfaces between the company ERP systems, the developed MES system contains modules supporting quality management, production surveillance, production scheduling, and long term planning. All these components have been fully tested and have been introduced to real manufacturing process . For the implementation of our current long term planner, a free third party linear programming based solver (LP PLANNER) was used, together with the communication and data transformation wrapper. The whole scheduling takes less than 1 second on a standard PC (with 28 days, 50 products, and 3 machines considered). This completely satisfies the performance requirements. The short-term scheduler has been fully developed at Gedas, s.r.o. One of the most difficult tasks was to make the decision concerning the granularity of the agents in the design phase. Only a small group of "heavy-duty" planning agents each capable of carrying out complex planning tasks has been designed. The key deployed agent concept in this case study is distributed planning (mainly on the different levels of planning granularity and different level of compliance to the imposed constrains) and negotiation among the agents when resolving the conflicts and mutual replanning interdependencies.

3.3. Agent-based supply chain management

While above we discuss the use of agent based planning on intra-enterprize level of the the factory decision making process, the concept of agent-based planning is equally good for planning and coordinating interaction among the various members of complex supply chains (such as procurement, logistics, virtual enterprize formation and supply chain management).

In ATG we have been investigating the use of multi-agent techniques supporting coordinated action in Request-based Virtual Organizations for dynamic multinational clusters of ERP/CRM value chain actors. The Request-based Virtual Organization (RBVO) is a special kind of Virtual Organization [36]. A formation of a RBVO is based on a negotiation between independent actors willing to cooperate. Individual actors are motivated to join the Virtual Organization to increase their business opportunities and to be able to participate on larger scale contracts. The individual value chain actors act as service providers mainly for consulting, software implementation, installation and customization, training and maintenance. The motivation is to find best suitable consortium of service providers to meet customer requirements such as cost, expected quality of service based on experience in industrial domain and appropriate ERP solution, geographical location or language.

The domain of RBVOs organizes multi-party interaction in the environments that are non-centralized and with flat organizational and heavily relies on strong multi-party involvement. A project cannot be implemented in isolation by a single actor, RBVO formation can be initiated by several actors simultaneously. An important property of this domain set is that it requires partial knowledge sharing.The actors in the environment are motivated to keep a substantial part of their private planning knowledge and resource availability information undisclosed.

The process of RVBO composition in competitive, semi-trusted environments has been solved by original contracting algorithm that is based on use of incrementally refined acquaintance models (IRAM) – the model that the actor is maintaining about potential collaborators [31]. In fully trusted competitive environments there are used classical approaches based on combinatorial auctions and contract net protocols. Individual requestors bid for different parts of RVBO, based on competencies and services required. The way the potential project is decomposed into particular services depends on costs and availabilities of services provided by

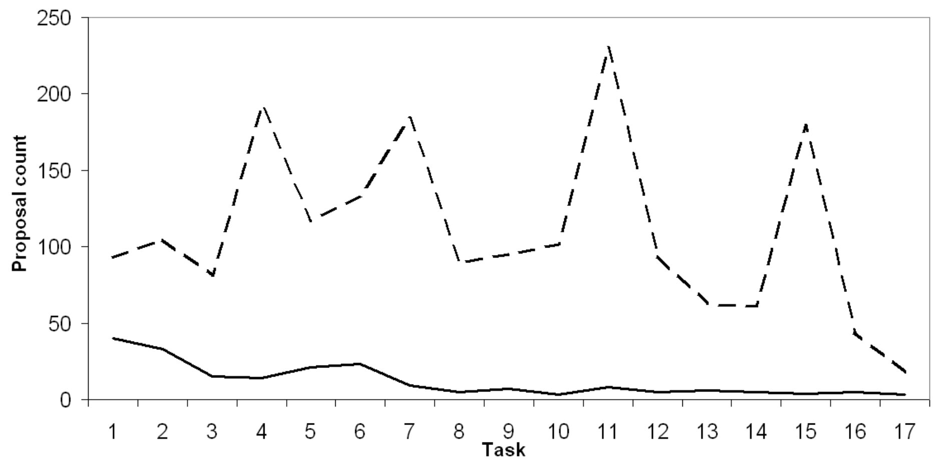

FIGURE 5. Amount of communication (with IRAM – solid line, without – dashed line.

different providers. If this information is unavailable (*uninformed decomoposition*), the requestor uses specific heuristic that guides a decomposition process (the minimal amount of RVBO members, balanced allocation of services, etc.). On the other hand, if the information about costs and services is available (*fully informed decomposition*), the requestor can evaluate each particular decomposition and select such that optimizes cost or deadlines.

The IRAM algorithm provides means to *partially informed decomposition*, typical for semi-trusted communities. There the bidders provides only partial information about their costs and availability. This partial information is used for building approximate acquaintance model, modeling the information used in fully informed decomposition. IRAM not only allows computing of an approximate decomposition, that is used for initiating one round of the contract-net protocol (seal-bid auction), but it can be also adapted should this round of negotiation terminate with a conflict deal[5] and improved for the next decomposition.

The IRAM algorithm has been tested on several RVBO formation scenarios. Empirically it has been validated that the IRAM algorithm (i) reduces the number of communication rounds and (ii) provides efficient tradeoff between the amount of disclosed information and quality of the resulting RVBO. This is documented e.g. on the Figure 5, that shows that total proposal communication by IRAM decreases with time, while fully informed decomposition requires substantially more communication in order to find an optimal RVBO.

[5]At least one of the bidder is not available to provide the service and thus the resulting RVBO would not be complete.

4. Conclusion

Multi-agent planning as defined in Section 2 represent important research challenge with a notable application potential. While there are strong research results in the communities of automated planning and multi-agent systems, given by their long-term tradition, the communication between these research subfield is limited. The community lacks formal models of the problem and abstract architectures with proven properties that can guide the multi-agent planning systems designers and developers.

In the presented paper we have challenged the only available general multi-agent planning architecture. Based on the well defined requirements for a multi-agent planning system, we came to a conclusion that the Durfee's architecture answers these challenges only in part. The requirements for non-centrality and multi-party involvement has been supported by subtask delegation and plan merging phases if done in decentralized manner. The presented architecture does not support in any way the requirements for fast replanning, real-time planning or opportunistic planning. While all the phases but delegation and plan merging are unaffected by interaction unavailability, the Durfee's planning architecture does not address this critical issue. The fact that the respective architecture cannot solve planning problems in adversarial environment is not surprising as the architecture has been designed primarily for cooperative environments.

Our view to multi-agent planning is component-oriented. We argue that the firm structure of decision making process as designed by Durfee can be generalized only to a limited extent. Therefore we propose the 4 (5) pivotal families of planning and agent-based computing methods from which a planning system ought by composed.

The automated planning component, of which we have provided a brief review in the paper, represent a centralized algorithm of any multi-agent planning system irrespective of the specialized requirements listed in the Section 2.1. The negotiation component enables non-centrality and multi-party involvement as much as it supports local interaction if some actors becomes temporarily inaccessible (requirement addressing for varying interaction availability). The acquaintance models component provide methods that address the same requirement in addition to coping with semi-trusted attitudes among the agents. The social commitments component makes the planning robust to changes. It needs to be said though that the use of social commitments presents additional requirement on the methods selected from the automated planning component. Slightly separate requirements for operation in adversarial environment is being handled by the adversarial reasoning methods.

In the paper we have also provided three different representative of multi-agent planning applications:

– **Agent based free-flight planning.** The presented system emphasized requirements for fast replanning and robustness against changes in the environment. As centralized collision free planning is a complex computational exercise and would prevent hardware deployment of the concept, the scenario also required decentralization and multi-party involvement. These requirements were addressed well by the variant of the monotonic concession protocol (an instance of the negotiation component) and the Accelerated A* planning algorithm (as an instance of automated planning component). The presented application cannot be implemented by the Durfee planning architecture as there is not subtask decomposition and delegation in this planning problem. The vehicles have their own private goals and objectives.

– **Agent-based production planning.** Our experience with production planning does not match with the challenges listed in Section 2.1. It can be implemented by means of a centralized planner (which was the original expectation of the sponsor). However, classical linear programming cannot have been used due to nonlinear properties of the problem specification and randomized planning methods did not provide good transparency for the users. The multi-agent application has been supported by a centralized planner (an instance of the automated planning component) on a relaxed specification of the problem complemented by a multi-agent simulation and conflict resolution process (an instance of the negotiation component).
– **Agent-based supply chain management.** In the field of supply chain management the application required the capability to deal with semi-trusted actors and with partially undisclosed data. Similarly, the requirement for varying interaction availability can play a role with small enterprize with unreliable connection to the Internet. Centralized approach as well as Durfee planning architecture were not suitable here. The used IRAM algorithm is an instance of acquaintance models based component and had been the core of the implemented application.

References

[1] M. Baioletti, S. Marcugini, and A. Milani. Dpplan: an algorithm for fast solutions extraction from a planning graph. In *5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS2000)*, Breckenridge, CO, USA, 2000.

[2] A. Blum and M. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, (90):281–300, 1997.

[3] W. Cao, C.-G. Bian, and G. Hartvigsen. Achieving efficient cooperation in a multiagent system: The twin-base modeling. In P. Kandzia and M. Klusch, editors, *Cooperative Information Agents*, number 1202 in LNAI, pages 210–221. Springer-Verlag, Heidelberg, 1997.

[4] A. I. Coles, M. Fox, and A. J. Smith. A new local-search algorithm for forward-chaining planning. In *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling (ICAPS 07)*, 2007.

[5] Mathijs M. de Weerdt, André Bos, J.F.M. Tonino, and Cees Witteveen. A resource logic for multi-agent plan merging. *Annals of Mathematics and Artificial Intelligence,*

special issue on Computational Logic in Multi-Agent Systems, 37(1–2):93–130, January 2003.

[6] Navin Bhat Department, Navin A. R. Bhat, and Kevin Leyton-brown. Computing nash equilibria of action-graph games. In *In Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI*, pages 35–42. AUAI Press, 2004.

[7] Marie E. DesJardins and Michael J. Wolverton. Coordinating a distributed planning system. *AI Magazine*, 20(4):45–53, 1999.

[8] Claude G. Diderich. A bibliography on minimax trees. *SIGACT News*, 24(4):82–89, 1993.

[9] Edmund H. Durfee. Distributed problem solving and planning. In Gerhard Weiß, editor, *A Modern Approach to Distributed Artificial Intelligence*, chapter 3. The MIT Press, San Francisco, CA, 1999.

[10] S. Edelkamp and M. Helmert. The model checking integrated planning system (mips). *AI Magazine*, 22(3):67–71, 2001.

[11] Ulle Endriss. Monotonic concession protocols for multilateral negotiation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 392–399, New York, NY, USA, 2006. ACM Press.

[12] Eithan Ephrati and Jeffrey S. Rosenschein. A heuristic technique for multiagent planning. *Annals of Mathematics and Artificial Intelligence*, 20(1–4):13–67, 1997.

[13] Klaus Fischer, Jörg P. Muller, Markus Pischel, and Darius Schier. A model for cooperative transportation scheduling. In *Proceedings of the First International Conference on Multiagent Systems.*, pages 109–116, Menlo park, California, June 1995. AAAI Press / MIT Press.

[14] M. P. Fourman. Propositional planning. In *Workshop on Model Theoretic Approaches to Planning, AIPS 2000*, Breckenridge Colorado, 2000.

[15] John Geanakoplos. Three brief proofs of arrow's impossibility theorem. *Economic Theory*, 26(1):211–215, July 2005.

[16] A. Gerevini, A. Saetti, and I. Serina. An approach to temporal planning and scheduling in domains with predicatable exogenous events. *Journal of Artificial Intelligence Research (JAIR)*, 25(187-231), 2006.

[17] Robert Givan and Thomas Dean. Model minimization, regression, and propositional strips planning. In *IJCAI*, pages 1163–1168, 1997.

[18] J.C. Harsanyi. Approaches to the bargaining problem before and after the theory of games: a critical discussion of zeuthen's, hick's, and nash's theories. *Econometrica*, (24):144–157, 1956.

[19] Nathanael Hyafil and Craig Boutilier. Regret minimizing equilibria and mechanisms for games with strict type uncertainty. In *In Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI-04*, pages 268–277. AUAI Press, 2004.

[20] R. M. Jensen, M. M. Veloso, and M. H. Bowling. In proc. of ecp01. In *2001*, OBDD-based optimistic and strong cyclic adversarial planning.

[21] H. Kautz and B. Selman. Unifying sat-based and graph-based planning. In *Proceedings of IJCAI-99*, Stockholm, 1999.

[22] Michael Kearns, Michael L. Littman, and Satinder Singh. Graphical models for game theory, 2001.

[23] Daphne Koller and Brian Milch. Multi-agent influence diagrams for representing and solving games. In *IJCAI*, pages 1027–1036, 2001.

[24] C. Li and M. Anbulagan. *Principles and Practice of Constraint Programming*, chapter Look-Ahead Versus Look-Back for Satisfiability Problems, pages 341–355. 1997.

[25] Viliam Lisý, Branislav Bošanský, Michal Jakob, and Michal Pěchouček. Goal-based adversarial search - searching game trees in complex domains using goal-based heuristic. In Joaquim Filipe, Ana Fred, and Bernadette Sharp, editors, *Proceedings of ICAART 2009 - First International Conference on Agents and Artificial Intelligence*, pages 53–60, Porto – Portugal, 2009. INSTICC (Institute for Systems and Technologies of Information, Control and Communication), INSTICC Press.

[26] Viliam Lisy, Branislav Bosansky, Roman Vaculin, and Michal Pechoucek. Agent subset adversarial search for complex domains. In submitted to: *Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS2010), YEAR = 2010,*.

[27] Carol Luckhart and Keki B. Irani. An algorithmic solution of n-person games. In *AAAI*, pages 158–162, 1986.

[28] V. Mařík, M. Pěchouček, and O. Štěpánková. Social knowledge in multi-agent systems. In M. Luck, V. Mařík, and O. Štěpánková, editors, *Multi-Agent Systems and Applications*, LNAI. Springer-Verlag, Heidelberg, 2001.

[29] David McAllester and David Rosenblitt. Systematic nonlinear planning. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 634–639, 1991.

[30] M. Pěchouček, V. Mařík, and O. Štěpánková. Role of acquaintance models in agent-based production planning systems. In M. Klusch and L. Kerschberg, editors, *Cooperative Infromation Agents IV - LNAI No. 1860*, pages 179–190, Heidelberg, July 2000. Springer Verlag.

[31] Michal Pěchouček, Vladimír Mařík, and Jaroslav Bárta. Role of acquaintance models in agent's private and semi-knowledge disclosure. *Knowledge-Based Systems*, (19):259–271, 2006.

[32] Michal Pechoucek, Martin Rehak, Petr Charvat, and Tomas Vlcek. Agent-based approach to mass-oriented production planning: Case study. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 37(3):386–395, May 2007.

[33] Michal Pěchouček and David Šišlák. Agent-based approach to free-flight planning, control, and simulation. *IEEE Intelligent Systems*, 24(1):14–17, Jan./Feb. 2009.

[34] Don Perugini, Dale Lambert, Leon Sterling, and Adrian Pearce. Agent-based global transportation scheduling in military logistics. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1278–1279, Washington, DC, USA, 2004. IEEE Computer Society.

[35] Michal Pěchouček; Jan Doubek; Jiří Vokřínek; Martin Rehák. Incrementaly refined acquaintance model (iram) for request-based virtual organization formation. In Vladimír Mařík; Jeffrey M. Bradshaw; Joachim Meyer; William A. Gruver; Petr Benda, editor, *Distributed Human-Machine Systems 2008*, pages 182–187. IEEE SMC, IEEE, 2008.

[36] Robert Roberts, Adomas Svirskas, and Brian Matthews. Request based virtual organisations (RBVO): An implementation scenario. In *Collaborative Networks and their Breeding Environments*, volume 186 of *IFIP*, pages 17–25. Springer, 2005.

[37] T. Sandholm. Contract types for satisficing task allocation: I theoretical results. In *Proceedings of the AAAI Spring Symposium*, 1998.

[38] B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In *Proceedings of the Second DIMACS Challange on Cliques, Coloring, and Satisfiability*, 1995.

[39] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* Cambridge University Press, New York, NY, USA, 2008.

[40] M. P. Singh, A. S. Rao, and M. P. Georgeff. *Multiagent Systems A Modern Approach to Distributed Artificial Intelligence*, chapter Formal Methods in DAI: Logic Based Representation and Reasoning, pages 201–258. MIT Press, Cambridge, MA., 1999.

[41] David Šišlák, Přemysl Volf, and Michal Pěchouček. Accelerated a* path planning. In Sierra Castelfranchi Decker, Sichman, editor, *Proceedings of 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, pages 1133–1134, Hungary, May 2009.

[42] R. G. Smith. The contract net protocol: High level communication and control in a distributed problem solver. *In IEEE Transactions on Computers*, C-29(12):1104–1113, 1980.

[43] H.-P. Störr. Planning in the fluent calculus using binary decision diagrams. *AI Magazine*, 22(3):103–106, 2001.

[44] Wittig T. *ARCHON: An Architecture for Multi-agent System.* Ellis Horwood, Chichester, 1992.

[45] A Tate, J Hendler, and M Drummond. A review of ai planning techniques. In J Allen, J Hendler, and A Tate, editors, *Readings in Planning*, pages 26 – 49, San Mateo, 1999. Kaufmann.

[46] Jiří Vokřínek, Antonín Komenda, and Michal Pěchouček. Decommitting in multi-agent execution in non-deterministic environment: Experimental approach. In *Proceedings of The Eight International Conference on Autonomous Agents and Multiagent Systems*, 2009.

[47] M. Wooldridge. *Reasoning about Rational Agents.* Intelligent robotics and autonomous agents. The MIT Press, 2000.

Michal Pěchouček – 2008 Curriculum Vitae

born June 29, 1972 in Kolin, Czech Republic, nationality Czech, married, three daughters

Present Affiliation

Associate professor at Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague (CTU),
Head of the Agent Technology Center, Deputy Head of Department, Head of Open Informatics Programme

Education

Engineering degree (eq. to M.Sc.) in Technical Cybernetics at Czech Technical University in Prague (1995)
M.Sc. in IT: Knowledge based systems, University of Edinburgh (1996)
PhD in Artificial Intelligence and Biocybernetics at Czech Technical University in Prague (1998)

Past Visiting Affiliations

- Visiting scientist at Artificial Intelligence Application Institute, University of Edinburgh, Royal Society of Edinburgh grant holder (2006)
- Visiting professor at State University of New York - University of Binghamton, lecturing a full semester course on Agent Technologies and Multi-Agent Systems (2003)
- Visiting postdoctoral researcher at the University of Calgary, Supported by Nortel Chair for Intelligent Manufacturing (2000)

Track Record in Research and Technology Transfer

Principal Investigator of the following research projects to Air Force Research Laboratory, NY AFOSR/EOARD:

- Agent-based Computing in Distributed Adversarial Planning, FA8655-07-1-3083 (2007-2008)
- Autonomous Agents for UAV Air-Traffic Control, FA8655-04-1-3044-P00001 (2005-2007)
- Modeling agents autonomy and agents' in adversarial environment, FA8655-04-1-3044 (2004-05)
- Meta-reasoning and Monitoring in the Multi-Agent Systems FA8655-02-M4056 (2002-2003)
- Agents Inaccessibility in Multi-agent Systems FA-8655-02-M-4057 (2002-2004)
- Acquaintance Models in Operations Other Than War Coalition Formation F61775-00-WE043 Multi-Agent Systems in Communication, investigator, F61775-99-WE099 (1999 - 2000)

Principal Investigator of the research projects to CERDEC US Army NJ:

- Reflective-Cognitive Adaptation for Network Intrusion Detection Systems, subcontract provided to Masaryk University (2008-2009)
- Distributed Planning and Coordination of Team-oriented Activities in a Dynamic Environment, W911NF-08-1-0041, subcontract provided to University of Edinburgh (2008)
- Distributed Planning and Coordination of Team-oriented Activities N62558-06-P-0353, subcontract provided to University of Edinburgh (2006-2007)
- Cooperative Adaptive Mechanism for Network Protection N62558-07-C-0001, subcontract provided to Masaryk University (2007)
- Modeling Individual, Collaborative and Adversarial Reflection in MAS N62558-05-C-0028 (05-07)

– Reflective/Cognitive Agent in Distributed Decision Making N62558-04-C-6001 (2004-2005)
– Modeling in Multi-agent Systems: A Technology Primer, co-pi, N62558-03-0819, (2004)

Principal Investigator of the research projects to Naval Research Laboratory, ONR/NRL:

– Meta-reasoning and Adjustable Autonomy in Computational MAS N00014-06-1-0232 (2005-08)
– Robot coordination using PIM, subcontract awarded by Florida IHMC, US N00014-06-1 (2007)
– Meta-reasoning for Modeling and Simulation in Multi-Agent Systems N00014-03-1-02 (2003-05)

Principal Investigator the following industrial research contracts

– BAE systems, Bristol, UK: Deployment of the AGENTFLY multi-agent system as a test-bed for reactive, probabilistic collision avoidance strategies (2007-2008)
– DENSO Automotive, GmbH: Agent based diagnostics in vehicle electronics (2005-2007)
– CADENCE Design Systems: A/GLOBE multi-agent system deployment for design process and assessment modeling (2006)

Local Coordinator of the European Commission FP5 and FP6 RTD Projects:

– CONTRACT - Contract Methods for Verifiable Cross-Organizational Networked Business Applications (2006-2008)
– PANDA - Collaborative Process Automation Support using Service Level Agreements (2006-2008)
– ExtraPLANT/EUTIST-AMI IST project, agent-based solution for supply chain management (03-04)
– ExPlanTech IST project, development of production planning multi-agent system (2000-2002)
– MPA GROWTH project, agent-based modular planning and simulation architecture (2002-2003)

Expert reviewer to EC in research and technology transfer projects (WIDE, SpiderWin, FluidWin – since 2004), Reviewer to projects funded by NOW – Netherlands Organisation for Scientific Research and IWT – the Research Funding and Innovation Stimulation Agency of the Flanders.

Principal Investigator of the subcontract provided by Institute for Human and Machine Cognition within the framework of HYRES project funded by NASA - development of an agent based root-cause detection, in hydrogen production facility (2003)

Coordination of the consulting project to GEDAS in cooperation with CERTICON, a.s. aimed the design of the agent based solution for engine manufacturing in SKODAAuto (2004)

Consulting to Rockwell Automation Research Center in Prague, design of the agent-based reconfiguration shipboard automation for the chilling system (2002)

PhD Committee member and an opponent on PhD thesis submitted to University of Edinburgh and Blekinge Institute of technology

Membership, Awards and Honorary Affiliation

Honorary/Visiting Member of Artificial Intelligence Application Institute, University of Edinburgh (since 2005)
Member of Advisory Board of Center for Advanced Information Technologies (CAIT), State University of New York (SUNY) in Binghamton (since 2005)
Member of the AgentLinkIII European Coordinating Action management committee, responsible for an industrial take up of agent technology (2004-2006)

The 2007 Engineering Academy of the Czech Republic Main Prize for AGLOBE multi-agent technology (for collective of authors)
The 2007 CIA (Cooperative Information Agents) Workshop Best Paper Award nomination (for collective of authors)

The 2006 DARPA Award for Best Industrial and Applied Paper at AAMAS 2006 (for collective of authors)
The 2005 Czech Technical University Chancellor Research Team Award to the Agent Technology Group,
The 2005 IEEE/WIC/ Intelligent Agent Technology Best Demo Award (for collective of authors)
The 2004 CTU Chancellor Award (3rd main prize) for excellence in industrial deployment of research
The 2004 CIA (Cooperative Information Agents) System Innovation Award (for collective of authors)
Best paper award at EMSCR 1998 (for collective of authors)
Siemens Dissertation Award 1998

Chairman of EUMAS AB (European Workshop on Multi-Agent Systems Advisory Board) (2004-2006)
AAMAS 2006 (Autonomous Agents and Multi-Agent Systems) senior program committee member;
AAMAS 2005/2006 (Autonomous Agents and Multi-Agent Systems) industry track co-chair; KSCO2002 (Knowledge Systems for Coalition Operation), KSCO 2004 co-chair; HoloMAS (Industrial Application of Holonic and Multi-Agent Systems) 2000 - 2002 co-chair; CEEMAS (Central and Eastern European Conference on Multi-Agent Systems) 2003, 2005 co-chair; program committee member of ESAW, ECAI (European Conference on Artificial Intelligence), CIA (Cooperative Information Agents), AAMAS (Autonomous Agents and Multi-Agent Systems),

Excalibur Alumni - Association of Czech Graduates of British Universities, founding member and chairman;
Member of CSKI –Czech Society for Cybernetics and Informatics (an ECCAI member);
Member of Academic Senate, Faculty of Electrical Engineering, Czech Technical University

Publications

List of publications and SCI references is available upon request. For selected publications see http://agents.felk.cvut.cz/publications/pechoucek

[illegible]