

Distributed Planning¹

Amol D. Mali, Subbarao Kambhampati

Arizona State University
{amol.mali,rao}@asu.edu

Distributed planning is the problem of finding a course of action that will help a set of agents in a given initial configuration to collectively satisfy certain desired behavioral constraints. The motivation behind distributed planning is to get the efficiency of parallel processing, the robustness of distributed systems and the simplicity of incremental construction and debugging. Problems that are inherently distributed (because of different spatial locations) or decomposable into subproblems with limited interactions are good candidates for distributed planning. Distributed planning may also be necessary when multiple agents (modules) must do independent but coordinated planning.

The dimensions along which distribution can take place are *world modeling*, *resources*, *replanning* and *functionality* (e.g. master-slave, planning-reacting etc.). In distributed planning, generally there is no global world model that can be looked at to know the state of the entire system. Instead, the model is distributed in the form of local world models. The resources that the agents manipulate, e.g. files in a software environment or the machines in a manufacturing environment, may be allocated to different agents. The problem solving expertise may be distributed as well. Some agents may do meta-level reasoning and assign low level tasks to other agents. These agents may ask other agents to execute the plans. If a plan is invalidated because of an environmental change, a new plan needs to be constructed. Such replanning may also involve multiple agents.

Distribution of planning activity generally makes communication necessary. In some environments, changes made to the world by one agent need not be explicitly communicated to other agents, because other agents can observe these changes in time, if they are constantly monitoring the environment. This is called “*communication through the world*”. However, not all changes are observable and some, such as changes in intentions, need to be informed explicitly. Agents either communicate successes (to enhance functionality) or failures (to seek help or prevent a chain reaction) or just the requested information. The agents that are involved in communication are either fixed *a priori* or decided at run time as per need. Harmful interactions among agents are resolved either locally (by negotiation or other protocols) or by a higher level agent dedicated to interaction resolution.

Currently, fully distributed planning systems do not exist (except those doing indirect distributed planning that we discuss later). Current systems either use partly local, partly global planning or completely global planning and

¹To appear in the Encyclopaedia of Distributed Computing, Kluwer Academic Publishers, 2003.

distributed scheduling (by planning we mean selecting relevant actions, while scheduling involves sequencing the selected actions). Execution is distributed in all these systems.

The planning activity can be either explicit or indirect. In explicit planning, the goals and subgoals are explicitly represented and actions are chosen to fulfill them. Indirect planning methods rely on cleverly designing behaviors of an agent to replace the explicit planning activity. The agents make decisions based on local information, however the decision making is set up in such a way that the agents fulfill tasks of interest. These agents are more reactive than the explicit planners. They use world as an external memory from which knowledge can be retrieved by perception. Hence the complexity of designing explicit planners is replaced by the complexity of designing agent-agent and agent-environment dynamics. The architectures of explicit and indirect distributed planning are shown in Figure 1. The dashed links and boxes there are optional and M, G denote meta-level information and goals respectively. In the following sections, we discuss various paradigms that the explicit and indirect planners follow.

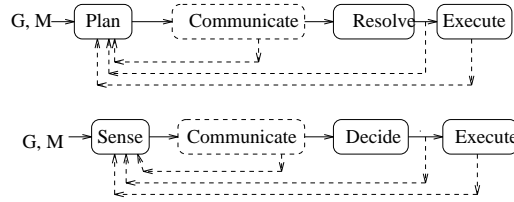


Figure 1: Architectures of Explicit and Agent-Environment Dynamics-based (Indirect) Distributed Planning

Explicit Distributed Planning

1. **Local Planning & Merging** - Agents independently synthesize plans and communicate them to avoid harmful interactions. Once individual plans are synthesized, they are merged after resolving harmful interactions. The distributed planning in Georgeff [6] is based on this paradigm. A supervisor process uses synchronization primitives to merge individual plans. The synchronization primitives are analogous to those in operating systems. Lansky & Fogelson [8] follow the same paradigm. They partition a domain into localized regions and a planner is assigned to each region. Plans from different regions are collected and interactions are resolved to yield the overall plan. The designer is responsible for exploiting the domain structure and arriving at the partitioning, e.g. the building construction domain can be partitioned using the physical structure of the building. A potential advantage of localization is that the interactions between localized regions are far fewer than all possible interactions that one has to consider in the absence of localization.

2. **Hierarchical Planning** - In the hierarchical planning process, plans are created at an abstract level and refined into less abstract plans, until an executable plan at the lowest level of abstraction is synthesized.

Corkill [4] discusses this type of planning in a distributed environment where the goal state is assumed to be a conjunction of many goals. Different goals are assigned to different processors. The world model is distributed among the processors. Critic is a mechanism that predicts or detects undesirable interactions, so that they can be corrected in time. This mechanism is also distributed among the processors. When the necessary information is not locally available, the processors communicate by broadcasting a request.

3. **Partial Global Planning** - Agents communicate their local plans to build plans that are partially global. These plans specify interactions among actions and avoid the myopic behavior resulting from purely local operation and the loss of real time functionality resulting from purely global operation.

Durfee & Lesser [5] propose this type of partial global planning for distributed problem solving. Their implementation of distributed vehicle monitoring involves interpreting data from different sensors at spatially distributed locations. The goal of the project is to create a global picture of vehicle traffic across the whole sensor net. For this purpose, each node has a separate partial global planner.

4. **Meta-level Control-based Planning** - Each agent in the distributed problem solving network is guided by a high level strategic information for cooperation among the network nodes.

Corkill & Lesser [3] follow this paradigm. The high level strategic plan that an agent uses is a form of meta-level control and it is represented as a structure that specifies in a general way, the information and control relationships among the nodes. The strategic plan lets an agent plan a sequence of its activities and adapt the sequence based on its problem solving role in the network. The strategic plan is somewhat similar to the partial global plan in [5]. However, partial global planners can modify the partial global plans and no such planner is used for for modifying the strategic plans.

Cammarata, McArthur & Steeb [2] follow a variation of this paradigm. They explore strategies of cooperation that agents may require to solve shared tasks effectively. A meta-level agent assigns roles to low level agents that synthesize their own goal-fulfilling behavior. This high level agent can also put constraints on the individual agents to eliminate undesirable interactions. The meta-level control here is centralized, whereas it is distributed in [3] in the form of separate strategic plans. Cammarata,

Mcarthur & Steeb [2] discuss cooperation strategies in the context of collision avoidance in air traffic control. While assigning roles to agents, the following heuristics are used by the meta-level agent - the agent that knows the most about other agents' actions is assigned the task of replanning, the agent that has more fuel, less aircrafts close by is chosen to do a more demanding task (since planning a path while avoiding more aircrafts is likely to take more time).

Indirect Distributed Planning

The agent-environment dynamics-based approaches can be classified into the following categories -

1. **Social Law-based Design** - A social law is a convention to be followed by agents in a group. A social law restricts the behavior of an agent. Social laws can reduce communication cost and planning time. For example, if agents navigate on a grid, we can restrict them to navigate only in their own row. As a result, collisions between agents from different rows are automatically avoided. However, it can be seen that the agents do not make any plans to avoid collisions, the social law is carefully designed to achieve this objective. Solutions found using social laws may not be optimal, but they may still be found faster than generative planning (synthesizing plans from scratch by composing action sequences).

Briggs & Cook [1] follow this paradigm. Since social laws can be too restrictive and limit soundness, they propose flexible social laws. A flexible social law is a set of laws with different degrees of restrictiveness. Hence the agents prefer to obey the laws, but can relax them (by choosing a less restrictive version of the law). Assuming a limit on the depth of search, each agent tries to plan with the strictest laws. If this fails, it chooses the next weaker law in the list. Choosing the strictest law is desirable because it provides the largest reduction in the search space of an agent. Agents however communicate whenever their actions interact. Since they provide multiple options, flexible social laws serve as naive anytime planning algorithms (an anytime algorithm is an algorithm whose execution can be interrupted at any time to get an answer).

2. **Computational Markets** - A market-oriented approach can be used to design and co-ordinate agents. The theory of general equilibrium provides a foundation for the construction of distributed planning systems based on price mechanisms. The activities of individual agents are defined in terms of production and consumption of commodities. The basic element of computation is an agent that has well scoped capabilities and a well-defined objective to maximize profits or utility. All interactions among agents are via trades at established prices, mediated by a uniform market mechanism.

Wellman [9] applies the computational market-based approach to distributed multicommodity flow planning, which involves allocating a given set of cargo movements over a given transportation network to minimize the cost. There are two types of goods - amount of cargo transported along a link and vehicles, fuel, labor. Carriers are agents of type *producer* who have the capability to transport cargo over specified links, given varying amounts of transportation resources. Each link has one carrier. Carriers submit bids specifying transportation services as a function of link prices and demand bids specifying required resources as a function of input prices. *Consumers* can buy, sell and consume goods. Their preferences for consuming various goods are specified by a utility function. An auction is associated with each distinct good. Agents submit bids to auctions. Given the bids from all interested agents, the auction derives a market-clearing price, at which the quantity demanded balances that supplied, within some specified tolerance.

Current Status

Despite the work described above, the field of distributed planning is in infancy. To examine the reasons behind this, we compare distributed planning with the related problem of classical planning that involves computing a sequence of actions to transform world from one state to another. Although there has been a lot of work on the classical planning problem in artificial intelligence, it assumes static and completely observable environments and perfect perception. Distributed planning systems tend to be open, where such assumptions do not necessarily hold. The perception may not be perfect and environments may be more dynamic and only partially observable. In a distributed environment, the planning problem also involves the problem of distributed knowledge representation, dealing with uncertain availability of resources and uncertain execution times of the actions.

Classical planning has focussed on efficiently synthesizing plans under simplifying assumptions and distributed planning research has focussed on synthesizing sound systems for operating in realistic environments. Because of many additional problems that classical planning does not subsume, efficiency has been a secondary issue in distributed planning. Can classical and distributed planning benefit from each other? A comparison of the research in these areas shows some possibilities for cross-fertilization. Recently, classical planners that do not work in the traditional “split & prune” style, frugally represent the space of all possible planning solutions in a disjunctive fashion and propagate constraints, have been shown to significantly speed up the planning process [7]. This progress however remains unexplored in the area of distributed planning. One can extend the disjunctive classical planners to compactly combine the plans of distributed interacting agents and dynamically add, delete and propagate constraints. One can transfer some ideas from distributed planning to classical planning as well, for example, classical planners can independently

plan for different goals and merge the plans efficiently as in Yang et al [10].

[1] Will Briggs and Diane Cook, Flexible Social Laws, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1995.

[2] Stephanie Cammarata, David Mcarthur and Randall Steeb, Strategies of cooperation in distributed problem solving, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1983, 767-770.

[3] Daniel D. Corkill and Victor R. Lesser, The use of meta-level control for coordination in a distributed problem solving network, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1983, 748-756.

[4] Daniel D. Corkill, Hierarchical planning in a distributed environment, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1979, Vol. 1, 168-175.

[5] Edmund Durfee and Victor R. Lesser, Predictability versus responsiveness: Coordinating problem solvers in dynamic domains, Proceedings of National Conference on Artificial Intelligence (AAAI), 1988, Vol. 1, 66-71.

[6] Michael Georgeff, Communication and interaction in multi-agent planning, Readings in distributed artificial intelligence, eds. A. Bond and L. Gasser, Morgan Kaufmann publishers, 1988, 200-204.

[7] Subbarao Kambhampati, Challenges in bridging the plan synthesis paradigms, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1997.

[8] Amy Lansky and David Fogelson, Localized representation and planning methods for parallel domains, Proceedings of National Conference on Artificial Intelligence (AAAI), 1987, 240-245.

[9] Michael Wellman, A market-oriented programming environment and its application to distributed multicommodity flow problems, Journal of AI research 1, 1993, 1-23.

[10] Qiang Yang, Dana S. Nau and James Hendler, Merging separately generated plans with restricted interactions, Computational intelligence, 8(4), Nov. 1992.