AI Magazine Volume 20 Number 4 (1999) (© AAAI)

Coordinating a Distributed Planning System

Marie desJardins and Michael Wolverton

■ Distributed SIPE (DSIPE) is a distributed planning system that provides decision support to human planners in a collaborative planning environment. The key contributions of our research on DSIPE are (1) constraint-based, consistent local views of the global plan that give each planner a view of how other planners' subplans relate to their local planning decisions; (2) methods for automatically identifying and sharing potentially relevant information among distributed planning agents; and (3) techniques for merging subplans that leverage the shared subplan structure to generate a complete, final plan. DSIPE is a fully implemented system and has been demonstrated to end users in the maritime (United States Navy and United States Marine Corps) planning community.

This article describes distributed SIPE-2 (system for interactive planning and execution) (DSIPE), a distributed version of the SIPE-2 hierarchical task network (HTN) planning system (Wilkins 1988). DSIPE provides the user with semiautomated generation of crisis-response options, in the presence of multiple, competing objectives and constraints, within a distributed computing environment that includes multiple planners collaboratively creating a plan to achieve a set of common objectives. DSIPE is most closely related to Corkill's (1979) distributed version of NOAH, a nonlinear hierarchical planner from which SIPE-2 is conceptually descended. DSIPE extends the ideas in distributed NOAH by focusing on efficient communication among planners and the creation of a common partial view of subplans. All the capabilities described in this article are part of the current implemented system, except where noted. Our current application domain is maritime campaign planning, but the techniques are domain independent.

Distributed Planning

In our distributed planning environment, multiple copies of the DSIPE planner run on separate processors that are connected across a network. These distributed processes communicate with each other by message passing.[1] Each instance of DSIPE supports a human planner. We refer to the human planner as the *user*, DSIPE as the *planning system*, and the user and planning system together as a *planning cell*.

As in SIPE-2, plans in DSIPE are represented as partially ordered networks of tasks and subgoals. Plans are expanded by applying *planning operators*, which represent templates or strategies for solving a goal. Each operator has preconditions (gating applicability conditions), constraints (used for binding variables and assigning resources), a plot (a partially ordered network of tasks and subgoals that expands the goal being solved), and effects (postconditions that are expected to be true in the world state after the plot is applied).

A high-level view of our distributed planning architecture is shown in figure 1. Goals are distributed from a coordinating planning cell down to lower-level planning cells, which exchange information with each other as they expand their subplans. (Although only two lower-level planning cells are shown, the architecture can support an arbitrary number of cells. Also, a lower-level planning cell can itself act in the role of a coordinator for additional planning cells, resulting in a hierarchy of cells.) Each planning cell has a partial view of the other cells' subplans, with explicit dependencies and relationships with the local cell's subplan.

Throughout the article, we demonstrate DSIPE's operation using an example from the maritime planning domain. The scenario is a noncombatant evacuation operation in which civilian personnel must be evacuated from a city where there has been an insurrection. A Marine Corps planner and a naval planner must work together to clear and land on a beach, move inland, and evacuate the civilians to an offshore vessel. The planners in this example have sepa-

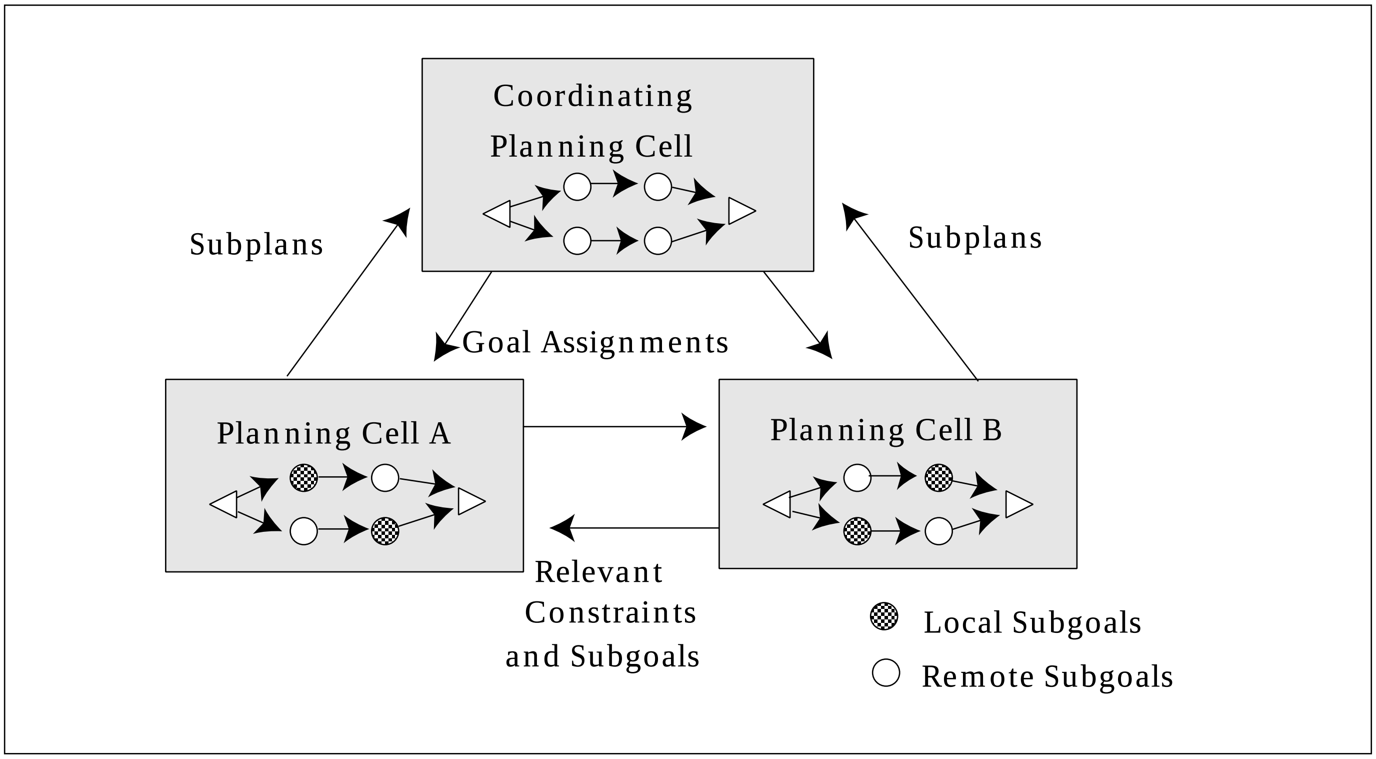

Figure 1. Distributed Planning Architecture.

rate but related requirements, goals, and priorities. Although their requirements overlap for this mission, they each have independent goals (other missions to be performed or supported), capabilities, and resources.

Distributing Goals

Distributed planning in DSIPE follows the hierarchical model used by SIPE-2. The coordinating planning cell develops a high-level plan that achieves the top-level objectives of the current planning problem. The coordinator then distributes this high-level plan to lower-level planning cells, partitioning the subgoals among the cells. This model works well in the military environment because the planning cell levels naturally correspond to echelons in the military hierarchy. In a less structured collaborative environment, alternative models of goal distribution might be required; for example, each cell might develop its own goal set and notify the other agents what it was planning to work on.

As shown in figure 2, the planning cells expand their subplans separately but share relevant information and constraints, as discussed in the following sections. At the end of the planning process, the coordinator merges the subplans together, as discussed in Plan Merging.

Figure 3 shows part of the Marine Corps subplan, before the planning cell has received any postconditions from the naval planning cell. The hexagons represent goals, and the rounded oblongs represent primitive tasks. The arrows show temporal orderings, and activities that are not ordered can occur in parallel. In this subplan, the Prep-Land (prepare to land on the beach) task is in parallel with another branch of the plan. In the lower branch, there are two goals in sequence: (1) Beach-App-Clr (clear the approach to the beach, that is, the water immediately adjacent to the beach, of mines and enemy troops) and (2) Returned-To (return to the offshore vessel). These two goals are in parallel with a third goal, Beach-Cleared (clear the beach itself). These two parallel branches are followed by several other goals: Trans-To (transport the marines from the beach to the evacuation site), Intel-Done (complete an intelligence survey of the evacuation site), and Evac-Grp-Done (transport the evacuees to a safe location).

The darker nodes in the display correspond to goals in the remote (naval) subplan. (These nodes appear in red in the screen display.) The other nodes are part of the local (Marine Corps) subplan. For example, the Prep-Land task and the Beach-Cleared subgoal are part of the Marine Corps subplan; the Beach-App-Clr and Returned-To subgoals belong to the Naval subplan.

Common Plan View

We extended SIPE-2's internal plan representation so that each planning cell has a complete representation of its own subplan as well as a partial representation of the subplans being developed by other planning cells. Subgoals being solved by other cells are represented in the local subplan as goal nodes that are not expanded. These nodes serve as placeholders for attaching postconditions and other constraints, which are sent by the remote planning cells that are responsible for expanding these goal nodes. The three types of constraint that are currently communicated among planning cells are (1) postconditions, (2) ordering constraints, and (3) variable constraints (including temporal variables).

During plan generation, postconditions are associated with tasks and subgoals through the process of operator expansion. Each planning cell automatically identifies postconditions that might be relevant to other cells, as discussed in the next section. Suppose the cell *A* determines that a set of postconditions are relevant to remote cell *B*. Suppose further that these postconditions appear in the expansion of a high-level goal *G* in *A*'s subplan. These postconditions are sent to cell *B*, which adds them to its subplan as additional nodes, appearing immediately before the placeholder node that corresponds to *G*. Along with the postconditions, ordering links and updated variable constraints are sent and attached to the added postcondition nodes.

The parts of a local subplan that a remote planning cell has been informed of make up a skeletal subplan that corresponds to the remote cell's view in the local cell's subplan. The local cell maintains a record of this skeletal subplan; that is, it keeps track of the partial ordering associated with the postconditions that have been sent to the remote cell. The ordering links associated with the skeletal subplan are sent to the remote planning cell, so that it can construct the same partial ordering for the postconditions. When additional postconditions are sent to the remote cell, only the ordering links necessary to incrementally update the skeletal plan are sent along with the postconditions.

All the constraints associated with variables in the postconditions are also passed to the remote agent. For example, if a postcondition is sent that states that unit *X* is located at point *Y*, the constraints on variables X and *Y* are also sent. These constraints can include class constraints (*X* is of type Force Reconnaissance Unit), properties (*X* has 40 troops), and instantiations (*Y* is bound to Pacifica-Beach-A).

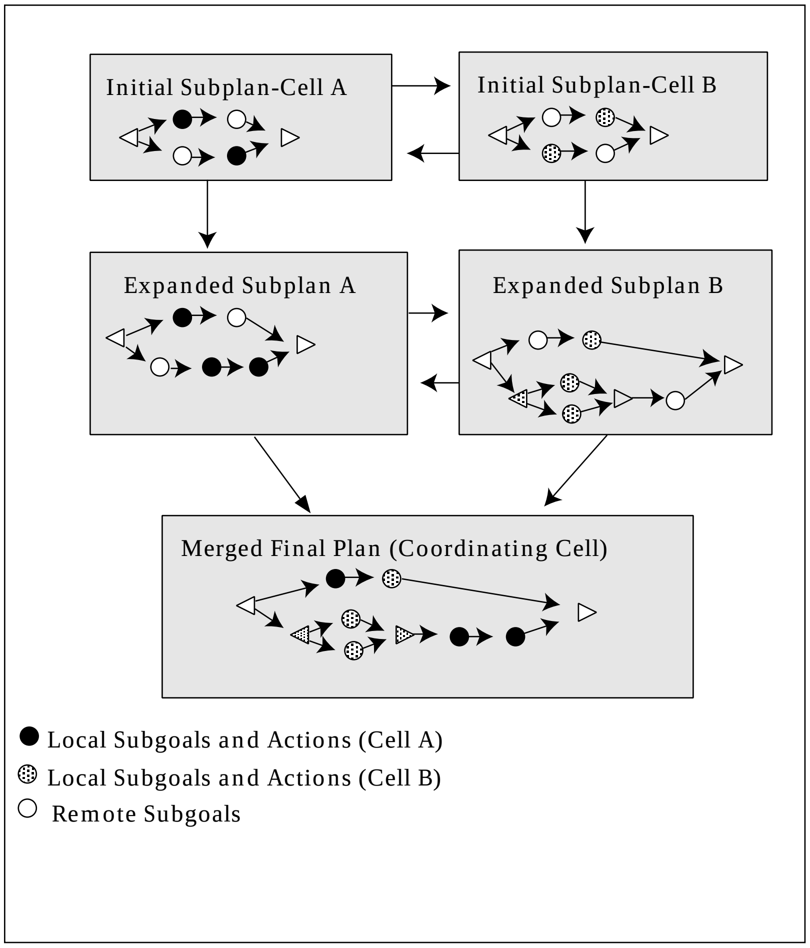
Initial Subplan-Cell A
Initial Subplan-Cell B
Expanded Subplan A
Expanded Subplan B
Merged Final Plan (Coordinating Cell)
● Local Subgoals and Actions (Cell A)
▨ Local Subgoals and Actions (Cell B)
○ Remote Subgoals

Figure 2. Marine Corps Planning Cell's Subplan before Receiving Naval Planning Cell's Postconditions.

The result is that each planning cell has a view of the overall plan that varies in detail: The view of the cell's own subplan is complete, but the view of the parts of the plan being developed by other cells are more or less sketchy, depending on how much information is determined to be relevant.

Information Sharing

The goal of information sharing in DSIPE is to minimize message passing among planning cells (especially important in limited-bandwidth environments) yet provide conflict detection and resolution early in the planning process. This goal is accomplished by sending constraints that arise during each cell's planning process to other cells that might need to use the constraints or that might generate subplans that conflict with them. Irrelevance reasoning is used to filter constraints that are guaranteed not to be of interest to other cells, thus reducing the communication requirements without sacrificing consistency.

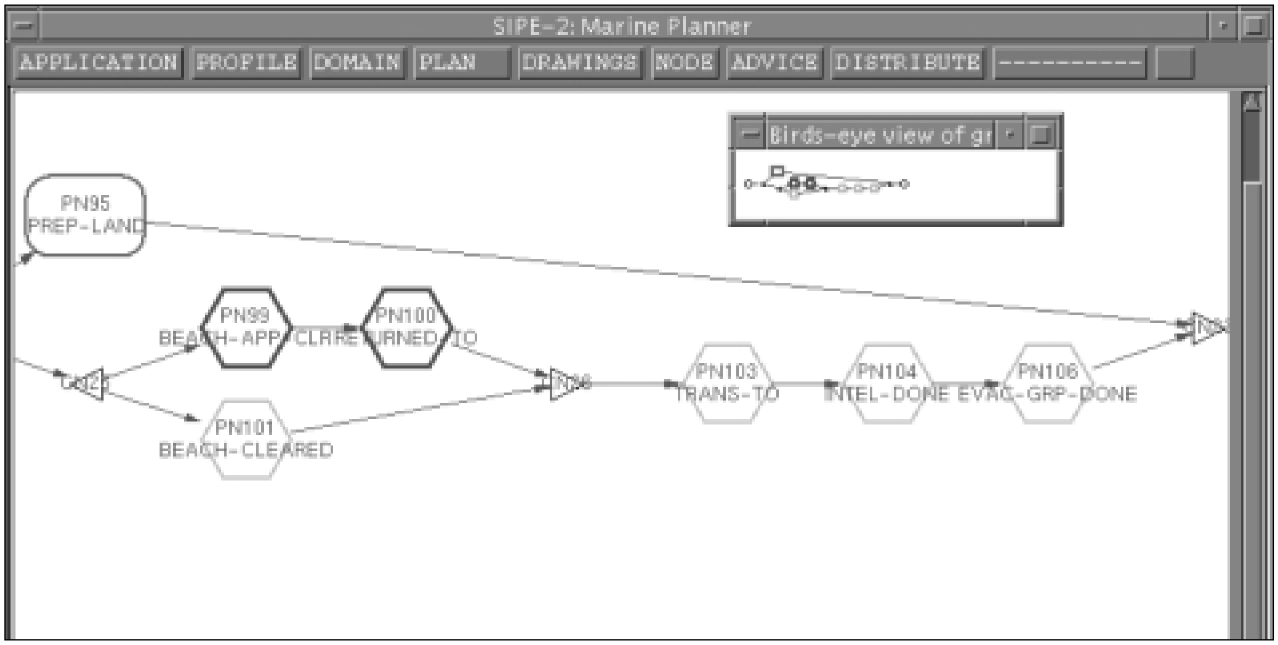

Figure 3. Marine Corps Planning Cell's Subplan before Receiving Naval Planning Cell's Postconditions.

Throughout the planning process, each planning system monitors the local cell's planning activity for constraints and subgoals that might be relevant to other planning cells and notifies the cells of this information. For example, the naval planning cell might notify the Marine Corps planner that a particular landing area will be swept of mines by a specified time. Currently, the only constraints that are monitored in this way are the postconditions. In future work, we will develop methods to monitor temporal and other variables, preconditions, and resource constraints.

Relevant preconditions are conditions that the local (sending) cell needs (or would like) to have the remote cell maintain or establish. For example, sending a precondition could tell the remote planning cell not to move a needed unit or to perform a surveillance activity for an intelligence estimate.

We implemented an algorithm for automatically filtering the postconditions that are sent from one planning cell to another by eliminating the postconditions that are provably irrelevant to the latter cell's decision-making process. The algorithm constructs a *query tree* (Levy and Sagiv 1993) for the objectives that have been assigned to a remote planning cell.[2] This tree can be constructed by the local cell if planning operators are shared by the cells or can be constructed remotely and sent to the local cell if each cell has its own set of planning operators.

The query tree identifies which predicates (that is, which types of knowledge) are used in the preconditions of the operators that might be applied to expand the objectives. The postconditions within the generated subplan are matched to the predicates in this tree to determine whether the postcondition might be relevant to the remote planner (that is, whether it potentially matches a precondition in the remote subplan).

Figure 4 shows a query tree for the Beach-Cleared goal in the maritime campaign planning example. The query tree is generated by identifying operators that could be used to achieve the Beach-Cleared goal, propagating any variable class constraints down the tree and applying the process recursively to goals identified within the operators in the tree. The preconditions associated with the operators in the resulting tree, together with the class constraints, describe the classes of ground facts that are potentially relevant to achieving the top-level objective(s) in the query tree.

As reported in Wolverton and desJardins (1998), irrelevance-based filtering can reduce message traffic substantially. For the two domains in which we ran experiments—a logistics domain and the maritime campaign planning domain presented here—using irrelevance-based filtering reduced the number of messages sent by 18 percent and 94 percent, respectively. Planning time was also reduced: Because fewer remote postconditions are added to the local subplan, less computation needs to be done to compute the impact of

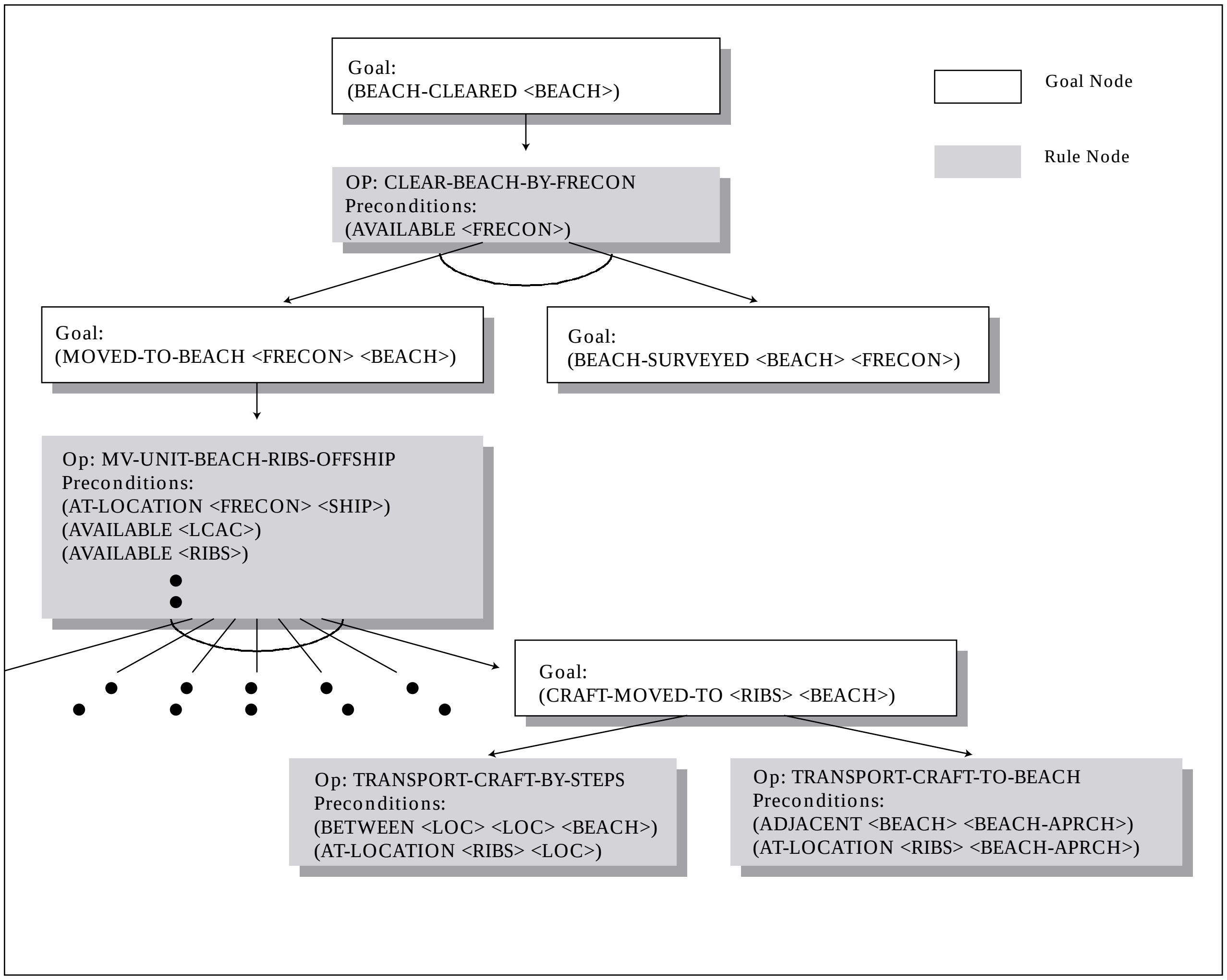
Goal:
(BEACH-CLEARED <BEACH>)
Goal Node
Rule Node
OP: CLEAR-BEACH-BY-FRECON
Preconditions:
(AVAILABLE <FRECON>)
Goal:
(MOVED-TO-BEACH <FRECON> <BEACH>)
Goal:
(BEACH-SURVEYED <BEACH> <FRECON>)
Op: MV-UNIT-BEACH-RIBS-OFFSHIP
Preconditions:
(AT-LOCATION <FRECON> <SHIP>)
(AVAILABLE <LCAC>)
(AVAILABLE <RIBS>)
Goal:
(CRAFT-MOVED-TO <RIBS> <BEACH>)
Op: TRANSPORT-CRAFT-BY-STEPS
Preconditions:
(BETWEEN <LOC> <LOC> <BEACH>)
(AT-LOCATION <RIBS> <LOC>)
Op: TRANSPORT-CRAFT-TO-BEACH
Preconditions:
(ADJACENT <BEACH> <BEACH-APRCH>)
(AT-LOCATION <RIBS> <BEACH-APRCH>)

Figure 4. Query Tree for the Beach-Cleared Goal.

these postconditions. Overall planning time was 35 percent lower in the logistics domain and 5 percent lower in the maritime planning domain.

During its planning process, the naval planning cell (not shown) identifies some potentially relevant postconditions in its subplan. In particular, it finds that some of the goals and tasks that were created during the expansion of its Beach-App-Clr subgoal might be relevant to the subgoals for which the Marine Corps planning cell is responsible. These postconditions correspond to the launching and movement of a small craft that the Navy will use to clear the approach to the beach. It sends these postconditions, along with ordering information that informs the Marine Corps planning cell of the order in which the postconditions should appear. Figure 5 shows the Marine Corps planner after these postconditions have been received and inserted into its subplan. The postconditions are encoded in the dark rounded rectangles that appear before the Beach-App-Clr goal.

After the Marine Corps cell has expanded its subplan further, several dependencies, constraining the partial order of the subplan, are introduced (figure 6). These ordering constraints represent dependencies between the naval planning cell's postconditions and tasks in the Marine Corps subplan. For example, the Marine Corps has determined that it can use the Navy's small craft to transport a group of marines to the beach. The Marine Corps subgoal to launch a craft (Craft-To, PM420) can be solved using the Navy's craft; the naval launch (Craft-TO, PM458) is then constrained to occur before the point in the plan where the Marine Corps requires the craft. The Marine Corps then introduces an additional Rendez action earlier in the plan to rendezvous with the Navy's craft for loading. With this dependency

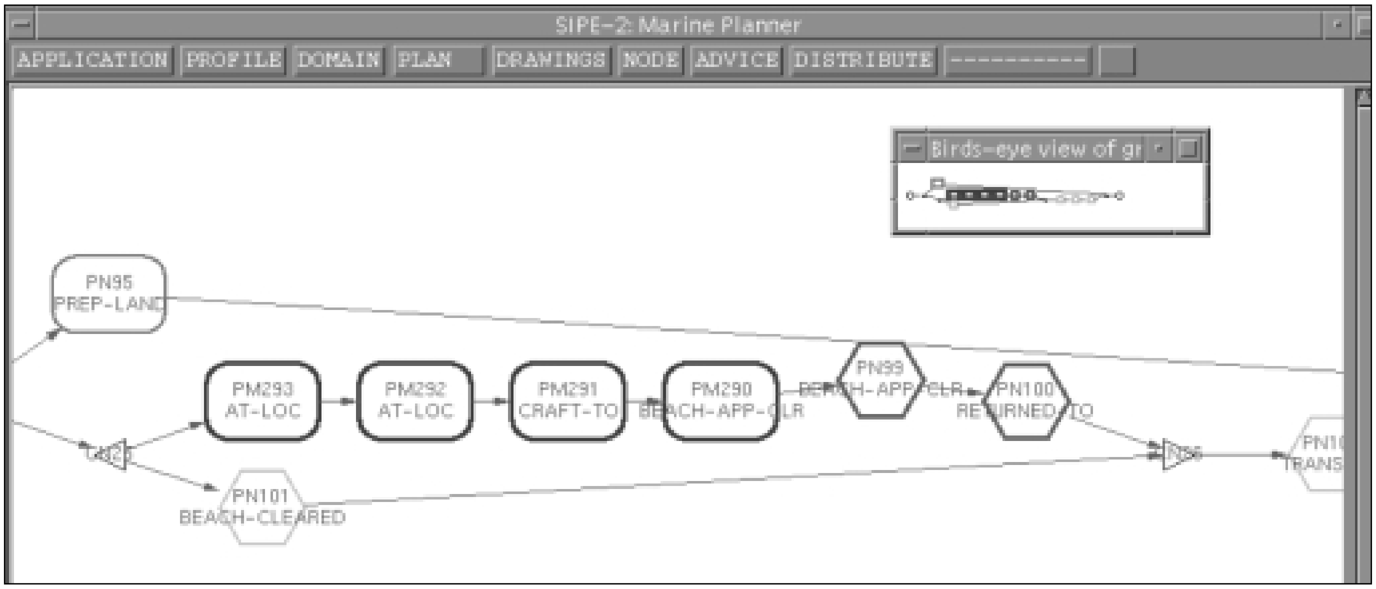

Figure 5. Marine Corps Planning Cell's Subplan after Receiving Relevant Postconditions (Dark Rounded Rectangles) from the Naval Planning Cell.

in place, it completes its planning, resulting in a subplan that is conditioned on the Navy's planned activities.

Plan Merging

After all the subplans are complete, they are submitted to the coordinating planning cell and merged. The shared plan structure (corresponding to the skeletal plans in each cell) guides the merging process, resulting in a merged partially ordered network (figure 7). SIPE-2's planning critics are then run to identify conflicts that were not previously detected.

Because the relevant constraints have been shared during the planning process, the expectation is that few, if any, conflicts will appear during plan merging. However, because of the complexity of planning dependencies, conflicts can arise. For example, although the temporal constraints between subplans are shared, propagation of temporal constraints among subplans is performed only when postconditions are passed from one cell to another. The result is that cycles of temporal constraints that are distributed across subplans might not be detected until plan-merging time.

If conflicts are detected during plan merging, in the current system, the coordinating cell resolves the conflict internally (for example, by repairing the plan or reallocating resources). In future work, we will incorporate methods for the coordinating cell to notify the subplanners to repair the conflict (see Conflict Resolution).

In the maritime planning example, the subplans are now complete and are submitted to the coordinating planning cell, which merges them. (In this case, the naval planning cell serves as the coordinating planning cell as well, but in the general case, the coordinating planning cell could be distinct from the other planning cells.) Figure 7 shows the merged plan. Although this display does not show the complete detailed plan, several parallel interactions can be seen, indicated by the three links from the upper part of the plan to the lower parallel branch. These interactions occur between the Marine Corps and the naval subplans and are all identified during the distributed planning process, then maintained when the plans are merged.

Future Work

The primary research directions that we are exploring are synchronization of planning cells, conflict resolution and negotiation, and improved information management. These areas are discussed in the following subsections.

Synchronization of Distributed Planners

One of the central research issues in DSIPE is how to communicate effectively about the planning process so that distributed planning cells can generate subplans on independent timelines, detect and resolve potential conflicts early, avoid duplicating effort, and create a high-qual-

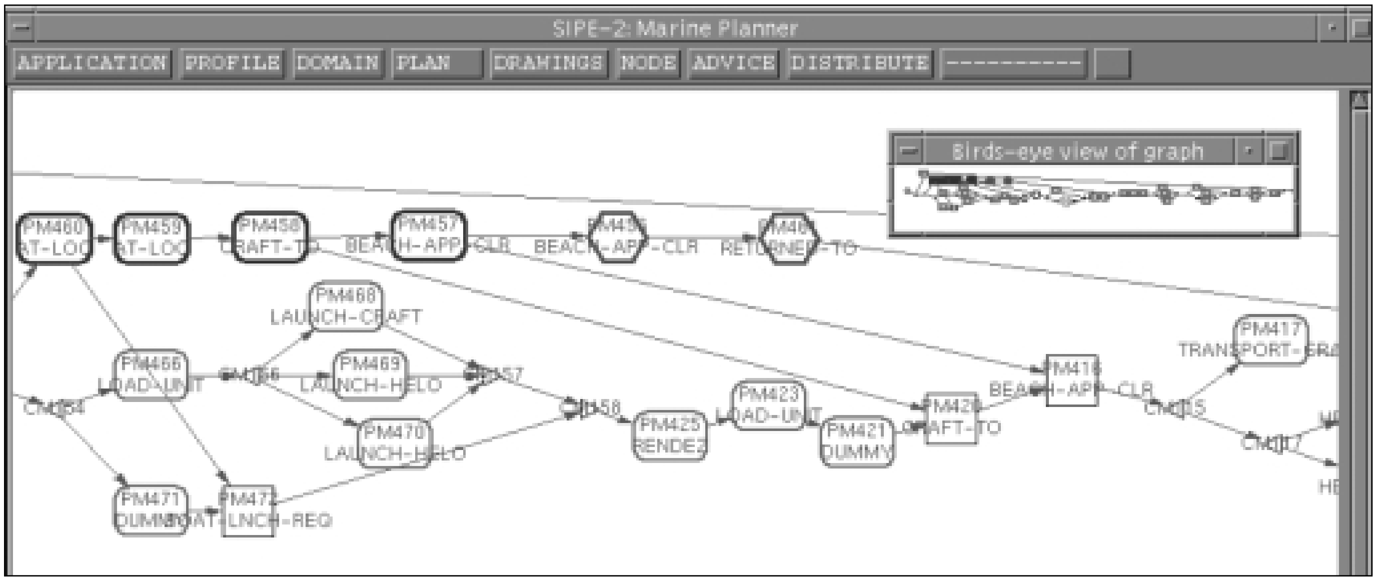

Figure 6. Interdependencies between the Marine Corps and Naval Subplans, from the Viewpoint of the Marine Corps Planning Cell.

ity merged plan at the end of the process.

The current distributed planning process is somewhat inflexible in that the primary way that planning cells share knowledge is by communicating postconditions that are created in one of the subplans. Also, variable constraints are only posted when potentially relevant postconditions are generated. Therefore, if one planning cell's preconditions depend on the postconditions of another cell, the part of the latter's subplan that produces the necessary postconditions must be completed first, so that the former cell can use the postconditions in constructing its subplan.

We are currently developing a more flexible process that will update information about variable constraints and instantiations dynamically as they change during planning. It will also allow other types of distributed constraint to be posted; for example, one cell could inform another cell of preconditions that it expected to be maintained or created in the world or could explicitly ask for assistance in achieving an objective.

We are also extending DSIPE's search-control methods so that subplans can be generated in different ways, depending on the situation. For example, a planning cell might decide to delay achievement of a particular objective (while it continues to plan out other objectives) pending more information about the other cells' subplans. To support this flexible search control, we will explore ways in which a planning cell can model remote subplans and planning processes in more detail.

Conflict Resolution

To date, we have focused on methods for representing and managing the distributed planning process and identifying constraint violations. However, making this technology robust and applicable to a broad range of distributed planning problems will also require developing techniques for resolving these conflicts.

The approach we are taking is to involve the user in the decision-making process and automate, as much as possible, those details of conflict detection and resolution that do not require human intervention. When a conflict is detected (or a potential conflict is predicted), either during distributed planning or at plan-merging time, the system will automatically analyze the conflict and determine a set of appropriate conflict-resolution strategies. These strategies can include replanning one or both subplans; agreeing to violate a precondition; requesting additional resources from the coordinating planning cell; or relaxing soft constraints, such as the arrival time of a unit. These strategies will be ranked and presented to the user, with information about the tradeoffs of the various strategies, to facilitate the decision making. If the conflict occurs during plan merging, additional heuristics will be invoked to decide whether the coordinating planning cell should repair the conflict on its own or whether the planning cells that generated the subplans involved in the conflict should collaboratively repair the conflict.

We are also exploring methods of opportunistic plan improvement where no con-

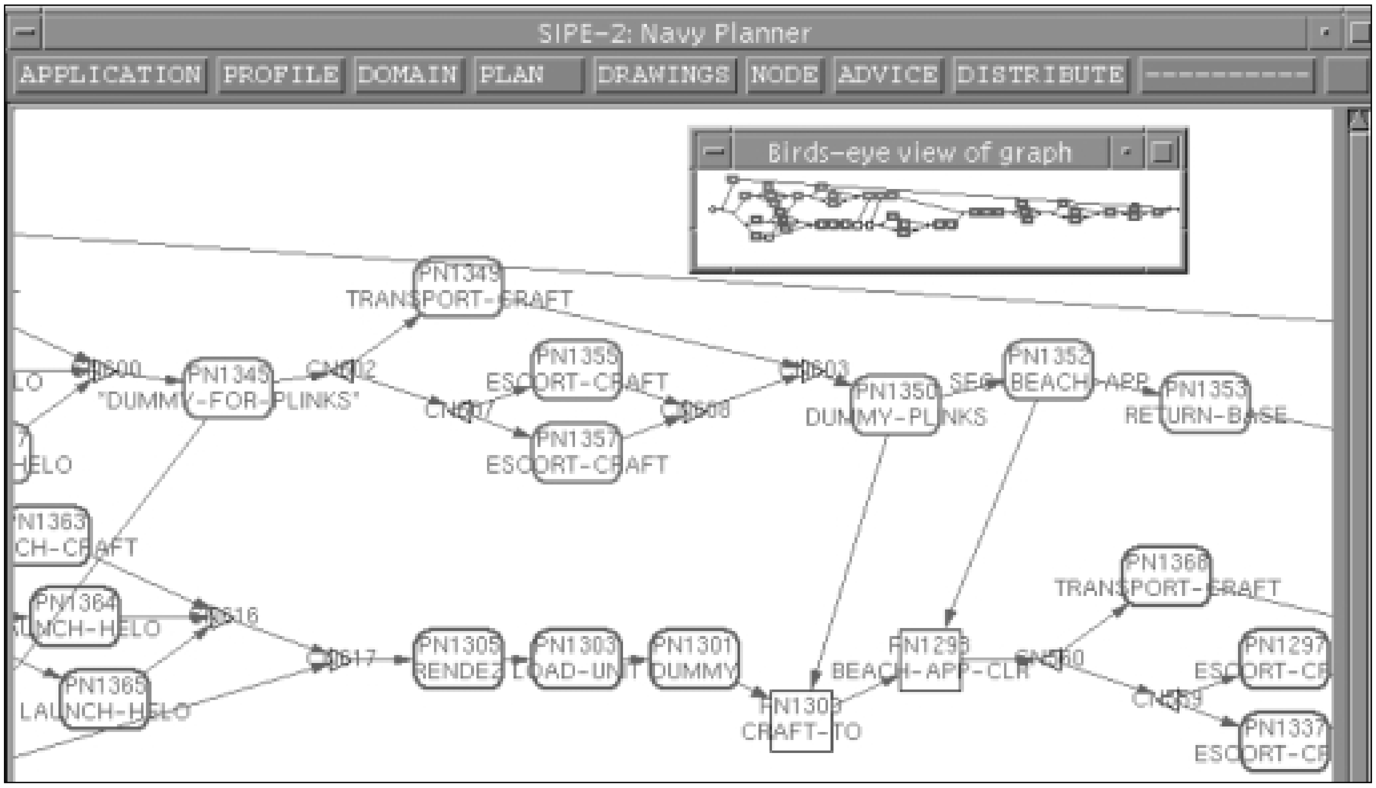

Figure 7. Final Merged Plan as Viewed from the Coordinating Planning Cell.

flict exists, for example, identifying and removing redundant parts of the plan or replanning to increase cooperation among the planning cells.

Information Management

The irrelevance-based filtering technique has been shown to reduce message traffic in DSIPE. The current implementation considers only the postconditions that are generated within each agent's subplan. The technique is general, however, and could be applied to other types of message traffic. For example, irrelevance reasoning could be used to determine whether to send requests for other agents to solve subgoals or requests to achieve or maintain preconditions. Another application of irrelevance reasoning would be to use the query trees to allocate subgoals among agents, based on some notion of locality, to minimize interactions among subplans. This approach is similar to how COLLAGE (Lansky and Getoor 1995; Lansky 1994) uses localization to decompose a planning problem.

Query trees could be pruned dynamically as subplans are generated (for example, when an alternative planning strategy has been ruled out, the associated preconditions would no longer be relevant). This approach would entail notifying remote cells of updates to the tree, so there would be a trade-off between the bandwidth required for the updates and the reduction in message traffic that would result from using the updated trees.

The irrelevance reasoning approach could also be extended by incorporating notions of probability and utility: How likely is it that an agent will in fact need to know a piece of information, and how useful will it be to the agent to know it? STEAM (Tambe 1997), a distributed agent architecture that extends joint-intentions theory (Levesque, Cohen, and Nunes 1990) to maintain a coherent view of the team's goals and plans, uses a decision-theoretic framework that incorporates the costs and benefits of communication as well as the probability that other agents already have the information in question. This approach, which is also used in Gmytrasiewicz, Durfee, and Wehe (1991), could be layered on top of DSIPE's irrelevance-based filtering to incorporate additional decision-making factors in deciding whether and when to communicate information between planning agents.

Conclusions

We described DSIPE, an implemented system that extends the state of the art of distributed

HTN planning. DSIPE provides decision support to human planners in a collaborative planning environment. The key contributions of this research are a shared plan representation, methods for automatically identifying and sharing relevant information among planners, and plan-merging techniques. We have also discussed important future research directions, including flexible synchronization of distributed planners, conflict-negotiation and conflict-resolution methods, and system evaluation.

Acknowledgments

This research was funded by the Office of Naval Research (ONR) and the Space and Naval Warfare Systems Command (SPAWAR) under contract N66001-97-C-8515. The authors would like to thank Mike Gherrity and Dave Swanson of SPAWAR for their guidance. Thanks also to David Wilkins for consulting and implementation assistance on SIPE-2 issues.

Notes

1. Our system supports messages passed using KQML (knowledge query message language) (Finin 1997) as text files or as CORBA-compliant distributed objects, using Xerox PARC's ILU implementation (Janssen et al. 1998).

2. The query tree is similar to an operator graph (Smith and Peot 1993). Previous planning systems used operator graphs to identify threats that can be deferred (Smith and Peot 1993) or further constrain the applicability of operators (Gerevini and Schubert 1996). In contrast, DSIPE uses the query tree to identify irrelevant classes of ground facts and prevent them from being considered by the planner.

References

Corkill, D. D. 1979. Hierarchical Planning in a Distributed Environment. In Proceedings of the Sixth International Joint Conference on Artificial Intelligence, 168–175. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Finin, T.; Labrou, Y.; and Mayfield, J. 1997. KQML as an Agent Communication Language. In *Software Agents*, ed. J. Bradshaw. Menlo Park, Calif.: AAAI Press.

Gerevini, A., and Schubert, L. 1996. Accelerating Partial-Order Planners: Some Techniques for Effective Search Control and Pruning. *Journal of Artificial Intelligence Research* 5:95–137.

Gmytrasiewicz, P. J.; Durfee, E. H.; and Wehe, D. K. 1991. The Utility of Communication in Coordinating Intelligent Agents. In Proceedings of the Ninth National Conference on Artificial Intelligence, 166–172. Menlo Park, Calif.: American Association for Artificial Intelligence.

Janssen, B.; Spreitzer, M.; Larner, D.; and Jacobi, C. 1998. ILU 2.0ALPHA12 Reference Manual. Technical Report, Xerox PARC, Palo Alto, California.

Lansky, A. L. 1994. Located Planning with Diverse Plan-Construction Methods. Technical Report FIA-TR-9405, NASA Ames Research Center, Mountain View, California.

Lansky, A. L., and Getoor, L. C. 1995. Scope and Abstraction: Two Criteria for Localized Planning. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, 1612–1619. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Levesque, H. J.; Cohen, P. R.; and Nunes, J. H. T. 1990. On Acting Together. In Proceedings of the Eighth National Conference on Artificial Intelligence, 94–99. Menlo Park, Calif.: American Association for Artificial Intelligence.

Levy, A. Y., and Sagiv, Y. 1993. Exploiting Irrelevance Reasoning to Guide Problem Solving. In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, 138–144. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Smith, D. E., and Peot, M. A. 1993. Postponing Threats in Partial-Order Planning. In Proceedings of the Eleventh National Conference on Artificial Intelligence, 500–506. Menlo Park, Calif.: American Association for Artificial Intelligence.

Tambe, M. 1997. Agent Architectures for Flexible, Practical Teamwork. In Proceedings of the Fourteenth National Conference on Artificial Intelligence, 22–28. Menlo Park, Calif.: American Association for Artificial Intelligence.

Wilkins, D. E. 1988. *Practical Planning: Extending the Classical AI Planning Paradigm*. San Francisco, Calif.: Morgan Kaufmann.

Wolverton, M. J., and desJardins, M. 1998. Controlling Communication in Distributed Planning Using Irrelevance Reasoning. In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), 868–874. Menlo Park, Calif.: American Association for Artificial Intelligence.

Marie desJardins is a senior computer scientist in the Artificial Intelligence Center at SRI International. Her current research projects focus on distributed planning and negotiation, machine learning, and information management. Other research interests include probabilistic reasoning, decision theory, and intelligent tutoring systems. desJardins received her Ph.D. in computer science–AI from the University of California at Berkeley in 1992. Her e-mail address is marie@ai.sri.com.

Michael Wolverton is a computer scientist in the Artificial Intelligence Center at SRI International. He received his Ph.D. in computer science from Stanford University in 1994. His current research focuses on planning, distributed AI, information management, and case-based reasoning. His e-mail address is mjw@ai.sri.com.

AAAI–2000 Robot Program Call for Participation

Austin Convention Center, Austin Texas ■ July 30-August 3, 2000

Sponsored by the American Association for Artificial Intelligence, the Office of Naval Research, and DARPA

We are seeking participants ... competition teams, exhibits, and challenge teams!

The Ninth Annual AAAI Mobile Robot Competition and Exhibition, held in conjunction with the AAAI National Conference on Artificial Intelligence, brings together teams from universities and other research laboratories to compete, and also to demonstrate cutting edge, state of the art research in robotics and artificial intelligence.

The mission of the Mobile Robot Competition and Exhibition is to serve AAAI, AI-robotics researchers, and the larger AI community by promoting innovative research through events which appeal to media and sponsors, while conducting these events in a format that facilitates comparison of approaches, but at low risk to individual or institutional reputations. Our goals are to:

- Foster the sharing of research and technology
- Allow research groups to showcase their achievements
- Encourage students to enter robotics and artificial intelligence fields at both the undergraduate and graduate level
- Increase awareness of the field

In previous years, the event has attracted both local and national news media — the 1996 contest resulted in a segment in Alan Alda's "Scientific American Frontiers" program on the Discovery Channel.

The Competition and Exhibition comprises three separate events; participants may enter any number of these events.

Contest

The contest allows teams from universities and other labs to show off their best attempts at solving common tasks in a competitive environment. Teams compete for place awards as well as for technical innovation awards, which reward particularly interesting solutions to problems. There will be two contest events this year. Details and rules are now available!

Exhibition

The exhibition gives researchers an opportunity to demonstrate state-of-the-art research in a less structured environment. Exhibits are scheduled through several days of the conference, and in addition to live exhibits, a video proceedings is produced.

Challenge

In addition to the contest and exhibit, we are adding a new aspect this year — the Robot Challenge. In this event, a particularly challenging task is defined which is well beyond current capabilities, will require multiple years to solve, and should encourage larger teams and collaborative efforts. The challenge task is defined by a long-term committee of researchers. Currently the task is for a robot to be dropped off at the front door of the conference venue, register itself as a student volunteer, perform various tasks as assigned, and talk at a session. The challenge will require integration of many areas of artificial intelligence as well as robotics.

Awards

Win a robot! Robots will be awarded for first and second place in two contest events! Contestants in contest and challenge events are qualified to win these prizes. Certificates of achievement and participation will also be awarded.

Robotics Workshop

On the last day of the conference, a robotics workshop will be held. Teams who receive travel support must attend and present at the workshop. All other participants will be strongly encouraged to attend and present. A research paper will be required within one month after the end of the workshop, and will be published in a proceedings. The purpose of this workshop is to allow researchers to understand and benefit from each others efforts.

Conference Proceedings

Each team will be allowed a two-page abstract in the AAAI Conference Proceedings. In order to have an abstract in the proceedings, you must register your team by March 15, and your camera-ready copy must be submitted by March 30.

Additional Information

For more information, please contact the chairs at the following address: aaai-robots@gmu.edu or visit the Robot Competition and Exhibition web site at www.aaai.org or www.aic.nrl.navy.mil/~schultz/aaai2000/

Participants are encouranged to join the aaai-robots moderated mailing list by sending email to "listproc@gmu.edu" and sending in the body of the message a request to join the list:

subscribe aaai-robots your-full-name-goes-here

Official notices will be posted to this mailing list.

Committee

General Chair: Alan C. Schultz (NRL)
Competition: Lisa Meeden (Swarthmore College)
Exhibition: Marc Bolen (CMU)
Challenge: Tucker Balch (CMU)
Workshop Chair: Karen Haigh (Honeywell)