SHALLOW PLANNING AND RECOVERY PLANNING BASED ON THE VERTICAL DECOMPOSITION OF THE FLIGHT DOMAIN

David C. Chen

E-Systems, Inc., Garland Division
P.O. Box 660023
Dallas, Texas 75266-0023

ABSTRACT

This paper presents a planning architecture that can perform both planning and recovery planning in the complex and dynamic flight domain. The key to such a robust planner is the vertical decomposition of the domain knowledge where the flight domain is decomposed into four minidomains, each a model of the flight domain along some degree of global viewpoint. The decomposition of the flight domain into four nearly independent minidomains and the explicit modeling of the different degrees of global viewpoint reduce the domain complexity geometrically. The vertical decomposition of the domain knowledge results in shallow planning and shallow recovery planning.

I INTRODUCTION

Recent planning works [1,2,3,4,5] have shown sophisticated approaches to make planning tractable for increasingly realistic and complex domains. This paper introduces a planning architecture designed for complex and dynamic domains such as the flight domain. In addition to the many aspects of flight such as routing and navigation, trajectory planning, piloting, and subsystems management, the complex and dynamic nature of the flight domain i6 manifested in the many domain constraints imposed on the flight or the plan. For example, a safe flight is a plan that satisfies domain constraints such as "the aircraft shall be navigable at all times," "the aircraft shall not run out of fuel," and "the aircraft shall not stall."

The difficulty in satisfying these domain constraints is that they have different scopes and they interact with each other. The scope of a constraint is the length of the plan over which the constraint holds, or how far the effects of the constraint span the plan. For example, the constraint "do not run out of fuel" has a very broad scope, from takeoff to landing, and the planner cannot determine whether the constraint is satisfied until the entire plan has been generated. On the other hand, the "do not stall" constraint has a short scope; it spans a small portion of the flight, on the order of one minute.

To complicate the situation further, the different aspects and the different constraints interact. For example, time efficiency dictates that the aircraft should cruise at high power setting, 75% power, which may add another refueling stop, increase the flight time, and possibly increase the flight distance. Since the "do not run out of fuel" constraint has a large scope, the planner cannot know that the 75% power setting, determined at the beginning of the flight, may cause a constraint violation until the end of the flight planning.

II VERTICAL DOMAIN KNOWLEDGE DECOMPOSITION

The planner can easily get into a catch-22 situation when trying to satisfy constraints of different scope. If the planner satisfies the narrow-scoped constraints first, it must guess at the global direction and do a tremendous amount of backtracking when the guesses do not satisfy the broad-scoped constraints. On the other hand, if the planner satisfies the broad-scoped constraints first, it must guess at the enablement precondition necessary to support the high-level plan; and again, much backtracking over the preconditions will occur. This catch-22 situation occurs because the planner tries to satisfy all the constraints at each instance during planning. Planning is much simpler if the planner only satisfies the constraints of one scope at a time. This is the essence of vertical domain knowledge decomposition.

A conceptual level is a minidomain that models the task domain at a breadth of scope. A conceptual level is a defined domain with its own world model, goals, operators, and constraints; it is a world unto itself. The task domain can be modeled at multiple scopes by constructing minidomains to match. The immediate effect of the conceptual levels approach Is to break the complex domain into simpler homogeneous minidomains, and consequently, complex planning is transformed into a series of shallow planning. This is similar to dividing a difficult task into easier subtasks and then solving the subtasks except the complexity reduction is achieved through domain knowledge partitioning instead of task decomposition.

The flight domain has been divided into four minidomains, the subsystems level and the three

This work was supported in part by NASA under contract no. NASA-NAG 1-288 and by Air Force under contract no. F49620-82-K-0009.

minidomains modeling flight at different scopes: the route level with the largest scope, the trajectory level of intermediate scope, and the aerodynamics level with the least scope. Within the subsystems level, planning knowledge is further partitioned into groups of the engine system, the fuel system, and the electrical system. This is an example of horizontal domain decomposition.

The route level models the flight domain along aspects of navigation and route selection. At the route level, the world is modeled as a network of nodes and links where the nodes represent the airports and the vortac navigational transmitters and the links represent navigable paths. The links are also the route-level operators. The trajectory level models the traversing of the three-dimensional space. The world here is the space defined by the x,y,z axis, and the goal is a specific point in space. The aircraft is again modeled as a point, except that it now moves along a trajectory in space. The trajectory-level operator is a vector in space. The aerodynamics level models how the aircraft flies through air. The aircraft in motion is modeled as a system of force vectors in equilibrium. Qualitative knowledge guides the manipulation of the force vectors to achieve the desired trajectory and the derivation of the discrete control settings. The exact settings for most analog physical actions such as the throttle setting are derived from the quantitative experts.

The difference in scope between two conceptual levels does not imply an abstraction relationship between the two levels. Nor is the representation of the broad-scoped level vague or fuzzy. To the route planner, the nodes and links are specific and sufficient to perform the routing task. Interestingly, physical actions can reside in any level. Selecting the vortac frequency and radial selectors is a natural part of the navigation task and should be a part of the route level. Flap and elevator controls are physical actions that belong at the aerodynamics level.

III LEVELS INTERACTIONS

In order for the system to function correctly, the minidomains must cooperate, forming a hierarchy. The hierarchy for the flight domain and the Interactions between the levels are shown in Figure 1. There are four kinds of interaction: the basis for the plan of the upper level is passed down to the lower level; the goal for the lower level is passed down from the upper level; the upper level may request the value of its capabilities from the lower level; and the capabilities of the upper level are updated by the lower level. These four interlevel actions are coordinated by the Interlevel planner.

Planning proceeds top down because it Is more efficient to satisfy the broad-scoped constraint first before satisfying the constraints of narrower scope. However, before the topmost planner can start, it needs to obtain Its current

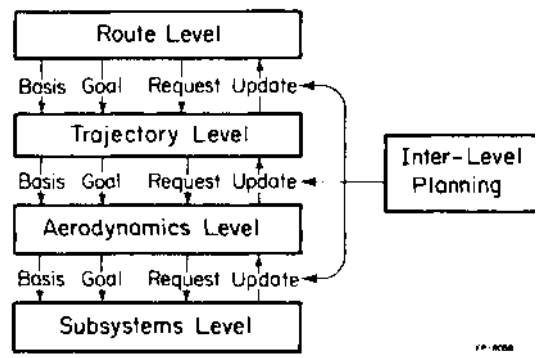

Figure 1 The Levels Hierarchy

capability. Model updating in this architecture is done bottom up because the lower levels have greater accuracy. For example, the route planner needs to know its current range and ceiling before it can start planning. The engine expert provides the engine thrust capacity. The aerodynamics planner calculates the aircraft ceiling based on the thrust capacity and the flaps and gear values. The ceiling is then passed upward to the route level. Figure 2 shows how the range is generated. The interlevel planner coordinates this process.

Although partitioning the task domain into minidomains may reduce the planning complexity at a given minidomain, it may also hide relevant information from the appropriate intralevel planner. For example, In order to streamline route planning, the aerodynamics knowledge is hidden from the route planner. However, the aircraft range varies greatly depending on how the aircraft is flown. A high power setting lowers the range. The range increases as the airspeed decreases and then decreases as the airspeed gets low. While the route planner should not be burdened with this knowledge, it should be able to benefit from this knowledge without any Inconvenience. This is done by allowing the route planner to request additional capability at the expense of some other capability. An example Is a request for an extra 20 miles of range with the airspeed negotiable from 250 knots to 200 knots. This request is depicted by the request arrow oi Figure 1.

IV RECOVERY PLANNING

The reader may see now that failure is a normal part of the planning process due to the partitioned, distributed planning architecture. Fortunately, because each minidomain is much smaller than the task domain, replanning is less complicated. A recovery planning expert can be devised for each of the four flight levels. For example, at the route level, if the destination airport shuts down, the planner backtracks, seeking the best refueling airport and forward-chains toward it. Plan failure can also

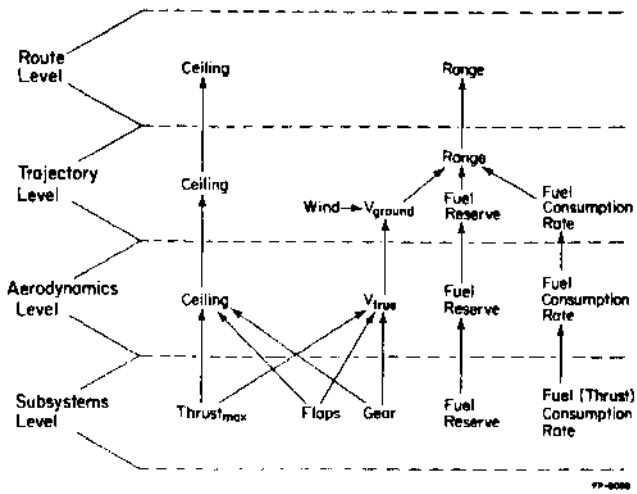

Figure 2 Bottom-Up Updating of the Minidomain Models

propagate across levels. For example, suppose a thunderstorm develops along the flight path. Since the trajectory planner cannot recover from this fault, it tells the route planner that the implementation of the airway segment has failed. Plan failure has now propagated to the route level. The route planner treats the thunderstorm fault as a link failure and patches around the broken link, making sure there is range to cover the added distance.

Recovery planning can be simplified if the basis Interlevel relationship is introduced. See Figure 1. When plan fails due to external changes, it is possible to recover locally within one level without disturbing the other levels. For example, suppose a headwind appears, changing the trajectory plan. The trajectory plan readjusts the landing phase. The trajectory planner knows that the range it gave the route level is no longer correct. However, if the trajectory planner knows the flight distance, it can determine if the route plan is still good. This is the purpose of the basis interlevel relationship in Figure 1. The basis is passed down from the upper level and tells the lower level what conditions must be true for the upper-level plan to hold. As long as the basis holds, the lower level can recover from plan faults without disturbing the upper level. This saves checking across the entire hierarchy every time the world changes.

V THE SOLUTION SPACE

Partitioning the domain knowledge reduces the solution space by funneling some interactions through bottlenecks at the boundaries. When the partitioning is based on the different scopes of viewpoint, the solution space reduction is dramatic, as shown in Figure 3. The pyramid on the left shows the solution space if planning is done at the ground level. The smaller pyramids on the right represent the solution spaces of the. levels. The importance of the vertical stacking of the four smaller pyramids is that the growth of the solution space is linear compared to the geometric growth of the pyramid on the left.

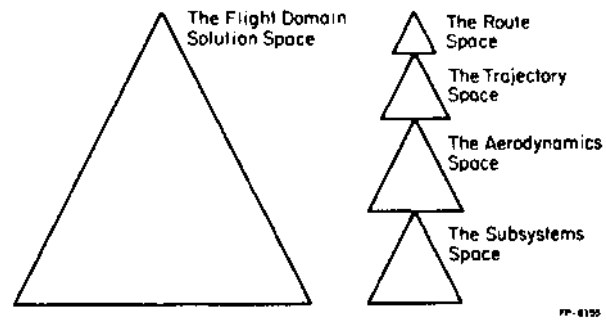

Figure 3 The Solution Spaces

VI IMPLEMENTATION

A flight planner called SECURE has been constructed based on the approach outlined in this paper. SECURE generates a sequence of control movements that flies a simplified aircraft between any pair of six airports and has demonstrated recovery planning for failures such as closed destination airport, tail wind and head wind, engine failure, thunderstorm, and stuck. Currently SECURE ignores near-ground operations and random turbulences.

ACKNOWLEDGEMENT

This paper is based on the author's Ph.D. thesis. The author is grateful to his thesis advisors, the late Professor R.T. Chien and Professor David Waltz.

REFERENCES

[1] Corkill, D. and Lesser, V. "The Use of Meta-Level Control for Coordination in a Distributed Problem Solving Network." In Proc. IJCAI-83. pp. 748-756.

[2] Friedland, P. "Knowledge-Based Experiment Design in Molecular Genetics." Report STAN-CS-79-771, Stanford University, 1979.

[3] Sacerdoti, E. "A Structure for Plans and Behavior." Tech. Note 109, SRI International, 1975.

[4] Stefik, M. "Planning with Constraints." Report STAN-CS-80-784, Stanford University, 1980.

[5J Tate, A. "Project Planning Using a Hierarchical Non-Linear Planner." Report No. 25, AI Research Dept., University of Edinburg, 1976.