

# Ordered Task Decomposition

**Dana Nau**

Dept. of Computer Science, and  
Institute for Systems Research  
University of Maryland  
College Park, MD 20742  
nau@cs.umd.edu

**David W. Aha**

Naval Center for Applied Research in AI  
Naval Research Laboratory  
4555 Overlook Ave. SW.  
Washington, DC 20375  
aha@aic.nrl.navy.mil

**Héctor Muñoz-Avila**

Dept. of Computer Science  
University of Maryland  
College Park, MD 20742  
munoz@cs.umd.edu

## Abstract

Ordered Task Decomposition is a new approach to HTN planning that appears to work quite well in a diverse set of real-world planning problems. This paper summarizes what Ordered Task Decomposition is, describes our past and current work on it, and describes our plans for future work.

## Introduction

Some of the requirements for a number of real-world planning tasks are shown in Table 1. Most action-based approaches to planning are limited by simplifying assumptions that sometimes fail to meet real-world requirements.

Examples of traditional action-based planning systems include Prodigy [Veloso and Blythe, 1994], causal-link planners such as UCPOP [Penberthy and Weld, 1992], planning-graph planners such as IPP [Koehler *et al.*, 1997], and satisfiability planners such as SatPlan [Kautz and Selman, 1996] and Blackbox [Kautz and Selman, 1999]. Table 1 lists some of the limitations of such planning systems with respect to real-world planning. For example, most of these planners can only deal with symbolic knowledge, which limits their applicability to planning problems that require mixed symbolic/numerical information-processing. Furthermore, the exponential time and space requirements of most of these systems limits their ability to scale up to complex planning problems. Hierarchical task network (HTN) planning systems (e.g., SIPE [Wilkins, 1990] and O-Plan [Tate, 1994]) have done a better job of addressing the requirements in Table 1, but they have tended to be large and complex systems that require a high degree of user expertise.

We have recently developed an approach called *ordered task decomposition* which overcomes many of these limitations. Ordered task decomposition is a variant of HTN planning that does a goal-directed search that proceeds forward from the current world state, planning each step in the order that it will later be executed.

## Relevant Previous Accomplishments

We have developed the underlying principles of ordered task decomposition, [Nau *et al.*, 1998; Nau *et al.*, 1999], and have successfully developed implementations of ordered task decomposition in several domains, as described below.

### The Game of Bridge

*Bridge Baron* is a computer program that plays the game of bridge. It uses ordered task decomposition to plan declarer play for bridge [Smith *et al.*, 1998a; Smith *et al.*, 1998b]. In about 90 seconds on average, it produces complete contingency plans for the possible moves that the opponents may make at each point in the game. Bridge Baron won the 1997 World Bridge Computer Challenge (as reported in *The New York Times* and *The Washington Post*), and thousands of copies have been sold commercially.

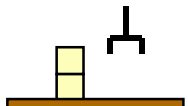
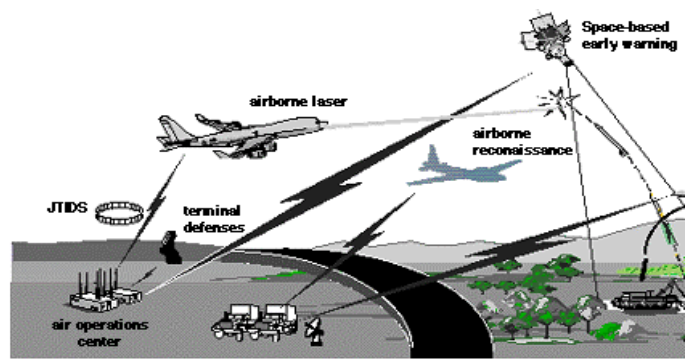
### Manufacturing Process Planning

The EDAPS system is an integrated system for design and manufacturing process-planning for microwave transmit/receive modules [Smith *et al.*, 1996]. By using the same approach (and some of the same code!) as in the Bridge Baron, it could generate process plans in only about one or two seconds. This work led to follow-up projects with Northrop Grumman, and results from this work have also been used in NIST's Process Specification Language (PSL) project [Schlenoff *et al.*, 1996].

### Domain-Independent Planning

The SHOP planning system [Nau *et al.*, 1999] is a domain-independent implementation of ordered task decomposition. A Lisp implementation is available at <http://www.cs.umd.edu/projects/shop> as freeware, and we are developing a Java implementation. SHOP has the following features:

Table 1: Typical restrictions of traditional approaches to planning, compared with the capabilities often needed for real-world planning.

	 <div style="border: 1px solid black; padding: 5px; background-color: #e0f0ff;"> <p><b>Unstack(x,y)</b>  <b>Pre:</b> on(x,y),  clear(x), handempty  <b>Delete:</b> on(x,y),  clear(x), handempty  <b>Add:</b> holding(x),  clear(y)</p> </div>			
Capability	Traditional restrictions	What is needed	Our approach	Our implementations
Scalability to large problems	Scalability limited by exponential space/time requirements	Fast enough to respond to users in real time on complex problems	Ordered task decomposition	Bridge Baron and EDAPS [Nau <i>et al.</i> , 1998]; SHOP [Nau <i>et al.</i> , 1999]
Embeddability	large monolithic systems	lightweight, embedded plan-inferencing tools	Implementation as embedded planner	Bridge Baron and EDAPS [Nau <i>et al.</i> , 1998]; HICAP [Munoz <i>et al.</i> , 1999a, 1999b]
Expressivity	Purely symbolic computations on logical atoms	Mixed symbolic & numeric computations on arbitrary data structures	Representational capabilities of ordered task decomposition	Bridge Baron and EDAPS [Nau <i>et al.</i> , 1998]; SHOP [Nau <i>et al.</i> , 1999]
Contingency planning	Perfect information; no contingencies occur	Imperfect information; need anytime access to contingency plans	Explicit parameterizable contingency plans using game trees	Bridge Baron [Nau <i>et al.</i> , 1998]
Multi-agent planning	The planner is the only agent capable of causing any change in the world	Need to model actions of other agents (both hostile and friendly)	Represent external agents' actions explicitly	Bridge Baron [Nau <i>et al.</i> , 1998]
Incremental retrieval	Plans generated from scratch; adaptation of previous plans not supported	Fast template retrieval and elaboration; incorporation of doctrine and prior experience	Hierarchical template memory, multiple indexing, retrieval by partial matching	HICAP [Munoz <i>et al.</i> , 1999a, 1999b]
Decision rationale	Reasoning only needs to be intelligible to researchers	Reasoning that is intelligible to users	Storage and explanation of decision rationale	(none yet)
Knowledge acquisition	Done manually	Reduce user effort for acquisition and maintenance	Semi-automated acquisition and maintenance	(none yet)

- **Simplicity:** the basic planning algorithm is only sixteen lines long, and the entire Lisp implementation is less than 1000 lines;
- **High representational power:** SHOP combines HTN decomposition, Horn-clause inference, and the ability to do numeric computations;
- **Computational efficiency:** in our tests on some standard benchmark problems, SHOP ran several orders of magnitude faster than other domain-independent planning systems.

### Military Operations Planning

In an ongoing collaboration on interactive plan authoring, we have embedded SHOP as the automated planning module in the HICAP system for authoring of Noncombatant Evacuation Operation (NEO) plans [Munoz-Avila *et al.*, 1999a; Munoz-Avila *et al.*, 1999b]. The basic structure of HICAP is shown in Figure 1.

HICAP dynamically elaborates plans, derived from military doctrine on NEOs and represented as HTNs (hierarchical task networks), via interactive case-base

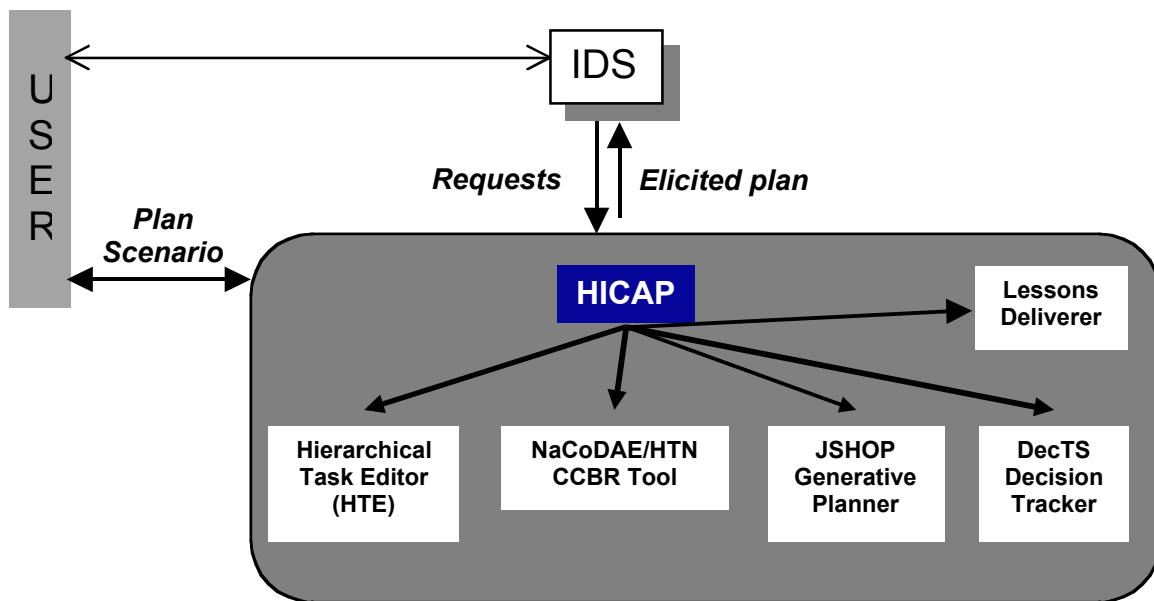


Figure 1. Architecture of the HICAP system for authoring of Noncombatant Evacuation Operation (NEO) plans.

inferencing [Aha and Breslow, 1997; Breslow and Aha, 1997]. HICAP assists users with dynamical plan elaboration by providing the following functionality:

- manual editing of plans represented using HTNs using a hierarchical task editor;
- interactive plan expansion using a case-based reasoning module called NaCoDAE [Aha and Breslow, 1997];
- automated plan expansion using JSHOP, our new Java implementation of the SHOP planning system;
- a lessons delivery module that, by monitoring HICAP's plan, can notify the user when lessons become applicable, and recommend corresponding plan elaboration operations [Weber *et al.*, 2000].

We are currently extending HICAP's capabilities as part of DARPA's Active Templates program [Dyer, 1999].

## Discussion

We believe that the following are the primary advantages of our ordered-task-decomposition approach:

- **A fast, simple, embeddable planning algorithm.** In contrast to most AI planning systems (which search backwards from the goal to reduce the amount of backtracking), backward search is unnecessary in ordered task decomposition, because the goal-directed focus can be maintained using the HTN decomposition itself [Nau *et al.*, 1998]. This produces a *significant speed gain*; for example, [Nau *et al.*, 1999] describes orders-of-magnitude speed gains versus "fast" AI planners. As described earlier, this approach has been successfully used to build several high-performance

embedded systems that are small and easy to use.

- **Expressivity.** At each step in generating a plan using ordered task decomposition, the planning algorithm has already planned for all preceding steps, so it can compute the current step's complete input state. Thus, it is not restricted to traditional plan representation techniques (where states and operator preconditions and effects are represented as sets of logical atoms, and are evaluated via unification), but can use arbitrary computer code to represent states, preconditions, and effects. This provides *more expressive state and operator representations*. For example, our implementations of ordered task decomposition (described earlier) perform mixed symbolic/numeric computation and inferencing, incorporate mixed-initiative interaction, perform "what if?" analysis and contingency planning, and incorporate case retrieval and modification.
- **The ability to do contingency planning.** In our application of ordered task decomposition to the game of bridge (described earlier), we used ordered task decomposition to generate and evaluate game trees that represent various contingencies that may arise in an imperfect-information game. We integrated this with a user interface for mixed-initiative interaction with users and external data input. The basic idea is to generate and retain alternative plan branches for likely violations to a plan's conditions (i.e., so that they can be quickly retrieved and executed), and to dynamically generate plans corresponding to unlikely contingencies.
- **Incremental retrieval techniques.** Incremental retrieval techniques have recently been developed in

the *conversational case-based reasoning* (CCBR) literature [Aha and Breslow, 1997; Munoz-Avila *et al.*, 1999a; Munoz-Avila *et al.*, 1999b] for data structures that are similar HTNs. We have adapted these techniques in order to incorporate JSHOP, the new Java version of our SHOP generative planner.

The primary limitations of our current work are as follows.

- Although we have implemented the abilities for contingency planning and for reasoning about external agents, we have only done so in a specific problem domain (the game of bridge). We still need to develop a domain-independent generalization and formalization of these techniques.
- Our proposed approaches for incorporating decision rationale and doing knowledge acquisition (see Table 1) have not yet been implemented.
- More work needs to be done to optimize the speed of SHOP's data structures, and to make it easier to debug domain descriptions in SHOP. As an example, we recently found that by making a simple modification to how SHOP represents its current state of the world, we could speed up SHOP by about an order of magnitude on large problems.

We intend to address the above problems in our future work, in order to extend the capabilities of SHOP and HICAP.

## Acknowledgements

This work has been supported in part by the following grants and contracts: ARL DAAL01-97-K0135, NRL N00173981G007, AFRL F306029910013, NSF DMI-9713718, and AFRL F30602-00-2-0505.

## Bibliography

D. W. Aha and L. A. Breslow. Refining conversational case libraries. In *Proceedings of the Second International Conference in Case-Based Reasoning* Providence, RI, 1997. Springer-Verlag.

L. A. Breslow and D. W. Aha. "NACODAE: Navy Conversational Decision Aids Environment." Tech. Report AIC-97-018, Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence, Washington, DC, 1997.

D. Dyer. "Active Templates: Dynamic Spreadsheets for Planning and Execution." Tech. Report, DARPA Information Systems Office, June, 1999.

H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings*

*of the 13th National Conference of the American Association for Artificial Intelligence*, pages 1194-1201, 1996.

H. Kautz and B. Selman. "Unifying SAT-based and Graph-based Planning". In *Proc. IJCAI-99*, Stockholm, 1999.

J. Koehler, B. Nebel, J. Hoffman and Y. Dimopoulos. Extending planning graphs to an ADL subset. In *Proc. ECP-97*, Toulouse, France, 1997.

H. Munoz-Avila, D. Aha, L. Breslow and D. Nau. HICAP: an interactive case-based planning architecture and its application to noncombatant evacuation operations. In *IAAI-99*, pages 870-875, 1999a.

H. Munoz-Avila, D. McFarlane, D. Aha, J. Ballas, L. Breslow and D. Nau. Using guidelines to constrain interactive case-based HTN planning. In *Proceedings of the Third International Conference on Case-Based Reasoning (ICCBR-99)*, 1999b. Finalist for the best paper award.

D. Nau, Y. Cao, A. Lotem and H. Munoz-Avila. SHOP: Simple Hierarchical Ordered Planner. In *IJCAI-99*, pages 968-973, 1999.

D. S. Nau, S. J. J. Smith and K. Erol. Control Strategies in HTN Planning: Theory versus Practice. In *AAAI-98/IAAI-98 Proceedings*, pages 1127-1133, 1998.

J. S. Penberthy and D. Weld. UCPOP: A Sound, Complete, Partial Order Planner for ADL. In *Proc. KR-92*, 1992.

C. Schlenoff, A. Knutilla and S. Ray. "Unified process specification language: Requirements for modeling process." Tech. Report, National Institute of Standards and Technology, Gaithersburg, MD 20899, September, 1996.

S. J. Smith, K. Hebbbar, D. Nau and I. Minis. Integrating Electrical and Mechanical Design and Process Planning. In *Proc. Second IFIP Workshop on Knowledge-Intensive CAD*, September, 1996.

S. J. J. Smith, D. S. Nau and T. Throop. "Computer Bridge: A Big Win for AI Planning." *AI Magazine*, 19(2):93-105, June, 1998a.

S. J. J. Smith, D. S. Nau and T. Throop. Success in Spades: Using AI Planning Techniques to Win the World Championship of Computer Bridge. In *AAAI-98/IAAI-98 Proceedings*, pages 1079-1086, 1998b.

A. Tate. Mixed Initiative Planning in O-Plan2. In *Proceedings of the ARPA/Rome Laboratory Knowledge-Based Planning and Scheduling Initiative*, pages 512-516. Tuscon, AR, 1994. Morgan Kaufmann.

M. Veloso and J. Blythe. Linkability: Examining causal

link commitments in partial-order planning. In *AIPS-94*, 1994.

R. Weber, D. W. Aha, L. K. Branting, J. R. Lucas and I. Becerra-Fernandez. Active case-based reasoning for lessons delivery systems. In *Proceedings of the Thirteenth International Conference of the Florida Artificial Intelligence Research Society*, Orlando, FL, 2000. AAAI Press. To appear.

D. Wilkins. "Can AI Planners Solve Practical Problems?" *Computational Intelligence*, 6(4):232-246, 1990.