# Hierarchical Goal Networks for Adaptive AI Systems

**Mark Roberts[1], Laura M. Hiatt [1], Dana Nau[2,3]**
**Sunandita Patra[2*], Ruoxi Li[2], Paul Zaidins[2], Morgan Fine-Morris[4], David Porfirio[4]**

[1]Navy Center for Applied Research in AI, U.S. Naval Research Laboratory, Washington, DC, USA
[2]Dept. of Computer Science and [3]Institute for Systems Research, Univ. of Maryland, College Park, MD, USA
[4]NRC Postdoctoral Research Associate, U.S. Naval Research Laboratory, Washington, DC, USA
[1]{first.last}@nrl.navy.mil | [2]{nau,patras,rli12314,pzaidins}@umd.edu | [4]{first.last}.ctr@nrl.navy.mil

Using goals is a hallmark of intelligent behavior (Simon and Newell 1971; Ram and Leake 1995; Cox and Zhang 2007; Hawes 2011; Shivashankar 2015; Aha 2018). In automated planning (e.g., (Ghallab, Nau, and Traverso 2016)), a goal is a partial world state an actor needs to achieve. An automated planner synthesizes a plan – a sequence of actions – to achieve the goal.

Hierarchical Networks are a type of plan representation that provides knowledge to advise a planner in solving a problem; knowledge is often in the form of *decomposition methods*. *Hierarchical Task Networks* (HTNs) are the more common form of this planning. HTN planning decomposes tasks (i.e., names) into subtasks (which are also names), which eventually decompose into actions. Because HTNs decompose task *names*, they are not strictly required to relate to the actual state of the world. For example, a cooking recipe describing how to prepare a dish does not necessarily list the required ingredients (they could be inferred) and may not relate the ingredients within a specific kitchen.

In contrast, Hierarchical Goal Networks (HGNs), purposely relate the decomposition methods to the world states – the goals – a system needs to complete (Shivashankar 2015). For example, the methods of an HGN would probably state that, for each ingredient, one must achieve the subgoal (i.e., the precondition) `have(ingredient)` before starting the recipe. HGNs are theoretically equivalent to HTNs (Alford et al. 2016) when one allows including HTN methods that link to world state, as done in some HTN systems. Thus, HGNs seem to offer no advantages. Why bother?

We argue that HGNs, and more broadly using ordered sets of goals, are preferable for use in Adaptive Systems for the following reasons:

- HGNs are a suitable abstraction for describing an outcome for a system to accomplish; it is relatively straightforward to specify goals and an ordering between them.
- HGNs are often a more compact representation for describing an outcome compared to specifying how to do it (i.e., the plan) because most plans are longer than the final goal. An HGN leaves under-specified the manner of accomplishment, allowing for execution flexibility.
- HGNs provide a clear way to explain what a system is doing or to explain its rationale. For example, an explanation could state "the robot achieved `opened(door)` to support `delivered(item)`".
- HGNs provide a natural bridge to developments in classical planning, planning landmarks, and to ongoing developments in heuristics from that literature.
- HGNs are amenable to formal methods; they provide a natural way to verify and validate that the system has met its specification because the sub/goals are both a state of the world that can be verified as completed (or maintained) and the state trajectory taken to complete the sub/goal(s) can also be verified.

We highlight three areas of work with collaborators related to HGNs, learning, and user interface design. A spanning theme is the use of goals or landmarks.

## 1 Adapting Hierarchical (Goal and Task) Networks for Execution

**Goal-Task Network and Implementation**: A common challenge for new practitioners of automated planning is learning the specialized language of a planning model (e.g., PDDL, ANML, SHOP, HDDL). One way to overcome this is to allow developers to use a common programming language, such as Python. Nau et al. (2021) developed a tool called GTPyhop (Goal-Task Pyhop) that can use goal or tasks and has been used in several projects mentioned in this paper. GTPyhop is a python based planner that uses an algorithm similar to SHOP (Nau et al. 2003); a key feature of this planner is that one only needs to write python to run the planner – no additional planning language is required. An iterative version of GTPyhop, called Iterative Pyhop (IPyhop), demonstrated a way to perform plan repair for robotics planning (Bansod et al. 2021). Recently this was improved by incorporating dependency detection and forward simulation to reduce future failures under plan repair (Zaidins, Roberts, and Nau 2023).

**Goal Lifecycle Networks**: A lifecycle describes the evolution of a process; a goal lifecycle describes how a goal progresses from formulation to completion. There are several well-known goal lifecycles in the literature. We presented a Goal Lifecycle Network (Roberts et al. 2021) that combines

*Work completed while a postdoc at UMD; Currently a Research Scientist at JPMorgan Chase.

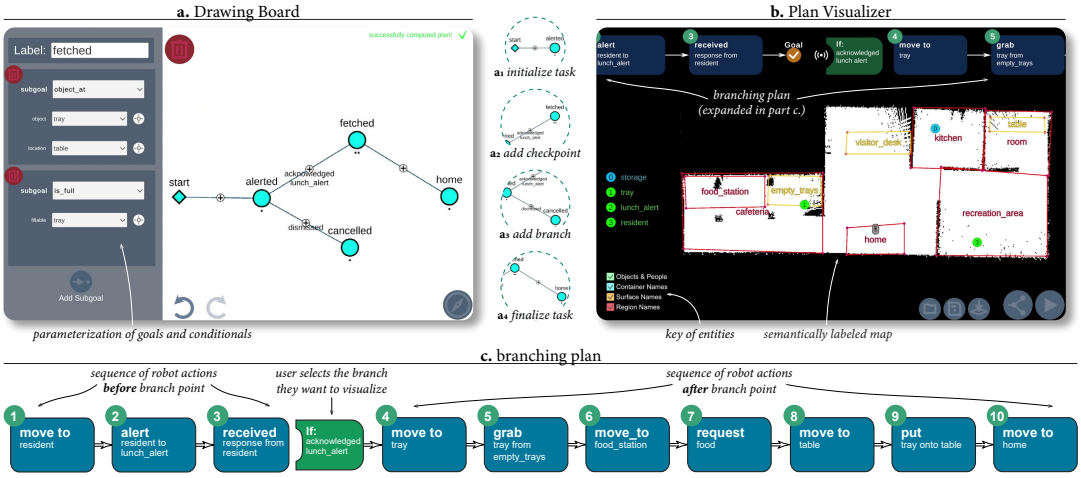


Figure 1: The Polaris user interface which includes (a) drawing board for specifying goal automata and (b) the PlanVisualizer. (c) The process of specifying the a branching plan corresponding to a delivery task.

HGNs with a goal lifecycle, linking it to a PDDL planner, and demonstrating its use in robotics. We demonstrated that goal networks were a useful tool for following the progression of goals through a robotic system.

**Goal Skill Network and Goal-Biased Curricula**: Most RL literature elicits behavior *implicitly* through the reward, though there are some recent initiatives that combine planning with RL or that provide abstractions similar to goals. Recent work by Patra et al. (2022) links a goal to a reward using a data structure called a *goal skill*, which they formalize as an RL Option with symbolic preconditions and effects. A collection of goal skills forms a *goal-skill network* (GSN), which allows allows a human expert to provide guidance for learning. Patra et al. (2023) extended the GSN work into a graph that partitions environmental and goal complexity, to design curricula for RL agents. The results showed that the task graph is a powerful conceptual tool for designing goal-based RL agents that combine planning and execution and that the task graph curriculum shows promise as a way to train goal skills.

## 2 Learning Hierarchical Planning Models

A challenge with using hierarchical models is the additional burden on the domain designer to create the decomposition methods. Instead they could be learned, which has a long history along with the literature on knowledge engineering (cf. the review by Callanan et al. (2022)). We briefly mention two lines of effort for learning hierarchical knowledge.

**Learning Numeric Models**: Fine-Morris et al. (2022) described a mechanism for learning numeric fluents in the context of hierarchical models. She showed that one can use natural language processing to locate bottlenecks in the domain, thereby approximating multi-problem domain landmarks. She leveraged this to construct methods that are aware of numerics. She has recently been exploring ways to incorporate text documents into a similar pipeline so that one could learn (or improve) planning models using knowledge and structure from written documents.

**Automatically Learning HTN Models using Landmarks**: One of the most well-known algorithms for learning HTN methods is called HTNMaker (Hogg, Muñoz Avila, and Kuter 2008). Li et al. (2022) developed a technique that uses planning landmarks and a curriculum to extend HTN-Maker so that it *automatically* learns hierarchical methods without the need for any human input. He has been extending this work to construct curricula from landmarks.

## 3 Integrating HGNs for End-User Development

The final thread of work involves an interface that allows users to instruct a robot using a variation of goal networks. End-User Development enables an "end-user" to "program" the robot without directly using a programming language or robotics framework. For example, a nurse in a caregiving situation may want a robot to perform tasks such as locating a patient in the facility and guiding that patient to their next activity. End-User Development is distinguished from robotics programming because the chief concern is in producing tools that facilitate ease of use, understandability for the end user, and a final robot "program" that performs the desired goals. Most of the work in EUD has focused on end users manually composing *plans* for the robot to execute.

In contrast, Porfirio et al. (Porfirio, Roberts, and Hiatt to appear) developed a platform called Polaris that allows an end-user to specify the desired *goals* of the system, which can then synthesize its own plans. Figure 1 shows the workflow in Polaris, whereby Fig 1(a) an end user creates and manipulates a goal automata, which is a variant of a goal network. Polaris converts this to a PDDL model and synthesizes a plan shown in the PlanVisualizer of Fig. 1(b). The user can specify branching plans such as the one in Fig. 1(c). We conducted a user study based on this tool and provided a detailed analysis showing where Polaris assists users in writing goal automata for robotic systems.

## References


Aha, D. W. 2018. Goal Reasoning: Foundations, Emerging Applications, and Prospects. *AI Magazine*, 39(2): 3–24.

Alford, R.; Shivashankar, V.; Roberts, M.; Frank, J.; and Aha, D. 2016. Hierarchical Planning: Relating Task and Goal Decomposition with Task Sharing. In *Proc. IJCAI*, 3022–3028. NYC, NY, USA.

Bansod, Y.; Nau, D.; Patra, S.; and Roberts, M. 2021. Integrating Planning and Acting With a Re-Entrant HTN Planner. In *ICAPS Wksp. on Hierarchical Planning (HPlan)*, 9.

Callanan, E.; De Venezia, R.; Armstrong, V.; Paredes, A.; Chakraborti, T.; and Muise, C. 2022. MACQ: A Holistic View of Model Acquisition Techniques. In *Proc. ICAPS*. ArXiv:2206.06530 [cs].

Cox, M. T.; and Zhang, C. 2007. Mixed-initiative goal manipulation. *AI Magazine*, 28(2): 62–62.

Fine-Morris, M.; Floyd, M. W.; Auslander, B.; Pennisi, G.; Gupta, K. M.; Roberts, M.; Heflin, J.; and Muñoz-Avila, H. 2022. Learning Decomposition Methods with Numeric Subtasks. In *Proc. of the Advances in Cognitive Systems*. Fairfax, VA.

Ghallab, M.; Nau, D.; and Traverso, P. 2016. *Automated Planning and Acting*. Cambridge University Press.

Hawes, N. 2011. A survey of motivation frameworks for intelligent systems. *JAIR*, 175(5-6): 1020–1036.

Hogg, C.; Muñoz Avila, H.; and Kuter, U. 2008. HTN-MAKER: Learning HTNs with Minimal Additional Knowledge Engineering Required. In *Proc. AAAI*, 950–956. AAAI Press.

Li, R.; Roberts, M.; Fine-Morris, M.; and Nau, D. 2022. Teaching an HTN Learner. In *Proceedings of HPLAN at ICAPS-22*.

Nau, D.; Patra, S.; Roberts, M.; Bansod, Y.; and Li, R. 2021. GTPyhop: A hierarchical goal+task planner implemented in Python. In *ICAPS Wksp. on Hierarchical Planning (HPlan)*.

Nau, D. S.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, J. W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *JAIR*, 20: 379–404.

Patra, S.; Cavolowsky, M.; Kulaksizoglu, O.; Li, R.; Hiatt, L.; Roberts, M.; and Nau, D. 2022. A Hierarchical Goal-Biased Curriculum for Training Reinforcement Learning. In *The International FLAIRS Conference Proceedings*, volume 35.

Patra, S.; Rademacher, P.; Jacobson, K.; Hassold, K.; Kulaksizoglu, O.; Hiatt, L.; Roberts, M.; and Nau, D. 2023. Relating Goal and Environmental Complexity for Improved Task Transfer: Initial Results. In *Neurips 2023 Workshop on Generalization in Planning*. New Orleans, Louisiana, USA. Citation Key: patraEtal23.genplan.relatingGoalEnvComplexity.

Porfirio, D.; Roberts, M.; and Hiatt, L. to appear. Goal-Oriented End-User Programming of Robots. In *Proceedings of HRI*.

Ram, A.; and Leake, D. 1995. Learning, goals, and learning goals, goal driven learning. *Art Intel Rev*, 9387–422.

Roberts, M.; Hiatt, L. M.; Shetty, V.; Brumback, B.; Enochs, B.; and Jampathom, P. 2021. Goal Lifecycle Networks For Robotics. In *Proc. FLAIRS*, volume 34.

Shivashankar, V. 2015. *Hierarchical Goal Networks: Formalisms and Algorithms for Planning and Acting*. Ph.D. thesis, Computer Science Dept., University of Maryland.

Simon, H. A.; and Newell, A. 1971. Human problem solving: The state of the theory in 1970. *American psychologist*, 26(2): 145.

Zaidins, P.; Roberts, M.; and Nau, D. 2023. Implicit Dependency Detection for HTN Plan Repair. In *Proceedings of HPLAN at ICAPS-23*.