Modeling Goal Selection with Program Synthesis

J. Branson Byers[1] **Bonan Zhao**[2] **Yael Niv**[1]
[1]Department of Psychology, Princeton University
[2]Department of Computer Science, Princeton University
{jbbyers,bnz,yael}@princeton.edu

Abstract

People can autonomously select and achieve novel goals to shape their own learning. But goal selection can involve selecting goals from large spaces, where repeated planning becomes computationally intractable. We propose program induction as an inductive bias for defining human-like priors to make goal selection easier. We demonstrate this tractable, semi-autonomous method for goal selection on a novel ShapeWorld task using a handcrafted grammar that maps states to reward functions.

1 Introduction

People can autonomously achieve new goals through *learning*. We can use signals of progress to learn how to cook unfamiliar foods (e.g. tasting while cooking) or to refine a unique art style (e.g. comparing to an imagined image). Goals like inventing new tools or coming up with (and testing) scientific theories can involve generative outcomes. Even when imagined outcomes have never been achieved or observed before, we can learn our way toward new feats using intrinsic evaluations of goal progress [1, 2, 3]. Such unsupervised learning is often modeled using the computational framework of reinforcement learning (RL) [4].

Goals support such directed learning processes. Goals shape how we represent actions, the environment, and the intrinsic rewards (e.g. signals of progress) that support problem solving [5, 6]. Goals also give humans – finite and rational agents – computational bounds on the problems we solve [7, 8, 9]. Selecting novel goals is a decision-making opportunity to choose how we learn, innovate, and grow. If RL can capture how people learn to achieve novel goals, how can we model how people select such goals in the first place?

Models of novel goal selection are rare because rational models of cognition typically capture decision-making with predetermined sets of choices [10], while the space of novel goals is often vast or unbounded [11]. The reason for considering limited choices is computational tractability. For novel goal selection, considering an unfamiliar objective can involve planning of how it may be achieved. While computationally expensive, this can reduce the uncertainty that comes with choosing a never-before-achieved goal [12]. Unfortunately, planning for each potential goal in a large space quickly becomes intractable.

Another hurdle for modeling goal selection is formalism. RL has no formal representation of a goal [13]. Instead, RL implicitly assumes that any goal can be represented using a Markov reward function R [14], a hypothesis that has been shown to have strict theoretical [15, 16, 17] and practical [18] limitations. Hierarchical, value-based models (e.g. options) circumvent this problem by casting goals as high level actions that consist of lower level routines [19]. While helpful in some contexts, these models do not capture how agents choose novel goals, since they rely on grouping existing routines to form new actions.

We propose program induction as a model for tractable selection of novel goals [20]. We first describe how a traditional, model-based algorithm would select novel goals, and then formalize program

Accepted at the Intrinsically Motivated Open-ended Learning workshop at NeurIPS 2024.

induction as an inductive bias to make this problem more tractable. We compare these methods on a novel goal selection paradigm, ShapeWorld, and conclude with future directions for this approach.

2 Framework

One challenge with selecting new goals is evaluating goal feasibility. It does not matter how rewarding a goal is, if that goal in unachievable. And if two goals have equivalent reward associated with their achievement, the easier to achieve goal should be preferred due to the external and cognitive costs associated with making more decisions [10]. Under these assumptions (and without loss of generality for variably-rewarding goal achievement), we consider a setting in which all goals have equal reward for achievement, but uncertain feasibility. To capture how agents might select goals that they do not yet know how to achieve, we force agents to choose novel goals before knowing their starting state. The intention is to model how agents might select novel goals when planning is difficult.

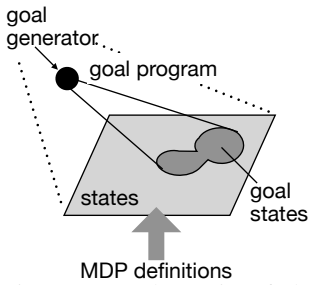

Figure 1: Schematic of the framework.

We therefore simplify the problem of goal selection to the problem of choosing easy-to-achieve goals under uncertainty. We review three different agents who solve this problem differently: 1) an optimal agent that uses value iteration to compute the value of each state as a potential goal, 2) a program-based agent that relies on program induction to choose good goals, and 3) an agent that chooses goals randomly. We then compare the strengths and weaknesses of these agents when it comes to selecting achievable goals.

Defining the goal selection problem. Consider a rewardless MDP $= (S, A, T, \gamma)$ where $s \in S$ are the states, $a \in A$ are the actions, $T(s, a, s')$ is the transition function between states, and γ temporally discounts potential rewards. We say that the goal state of the agent is s_g such that:

$$R(s) = \begin{cases} 1 & \text{if } s = s_g, \\ -1 & \text{otherwise.} \end{cases} \tag{1}$$

Thus, the objective of a reward maximizing RL agent is to select some s_g that can be achieved in the least number of (-1 costing) steps. This reflects the costs (external and cognitive) associated with pursuing a goal. We let the starting state $s_0 \in S$ be initialized at random *after* goal selection has occurred. This prevents the agent from trivially choosing an adjacent state as its goal.

Optimal goal selection through planning. Given randomly initialized starting states, an optimal agent must marginalize over all starting states to calculate which goals can be achieved in the least number of steps on average. In order to determine which goals are most achievable on average, we first calculate the optimal value function V^* under each goal using value iteration [4]. Since starting state s_0 is chosen at random, the value of choosing a particular goal state is the expected value of that goal, V_g, across all states:

$$V_g(s_g) = \sum_{s \in S} V^*(s|s_g)p(s) \tag{2}$$

Goals can then be selected greedily, or using softmax:

$$P_{RL}(s_g) = \frac{e^{V(s_g)}}{\sum_{s'_g} e^{V(s'_g)}} \tag{3}$$

Program-based goal selection. We define a goal as an ordered pair $g = (z_g, R)$, where z_g is a compact goal embedding that we call a "goal program,"[5, 21]. Goal programs are sampled from a generative grammar G, which assigns a probabilistic distribution over a potentially infinite space of goal programs [21, 22, 23]:

$$z_q \sim G \tag{4}$$

Crucially, a goal program z_g generated by G corresponds to a set of states in the MDP (Figure 1). We can say the set of states S_z to which the program applies are *covered* by z_g. Thus we can think of a

goal program as a binary function over states. We then sample $s_g \sim Uniform(S_z)$ to define a reward function as in (1). While in theory a goal program could cover and reward many states, allowing multiple states to satisfy a current goal would make programs with large coverage more feasible. We constrain the kinds of goals the agent can set to single-state goals for simplicity. This preserves the assumption that all goals are created equal, save for the risk described by T and uncertainty over s_0. We leave inspecting the relationship between program coverage and goal selection to future work.

Though not necessary for goal selection, we can compare the predictions of the optimal agent P_{RL} to those of a program-based agent P_G. This can be done by calculating the likelihood a state s is covered by a program z_g generated by G. Let $P_G(z_g)$ be the probability that z_g is chosen from G and $P(s|z_g)$ be the probability s is sampled from the states covered by z_g. Then:

$$P_G(s) = \sum_{z_g \in G} P_G(z_g) P(s|z_g). \tag{5}$$

Random goal selection. While agnostic to the objective of selecting easy to achieve goals, a random agent can still choose a goal state s_g without taking into account any information about the environment, and try to then maximize reward according to Eq (1):

$$s_g \sim \text{Uniform}(S). \tag{6}$$

Advantages and Disadvantages. The optimal agent uses value iteration to compute the optimal value function $V^*(s|s_g)$ for every $s_g \in S$. This means repeatedly solving the RL problem for each state. While choosing greedily using V_g will select the easiest to achieve goals (e.g. goals with the highest utility), in large state spaces it is prohibitively expensive to compute V_g for every possible goal.

By contrast, a program-based agent samples a program z_g from G and then a state s_g covered by z_g. This reduces the computational cost of choosing a goal by putting the onus of good goal selection on the grammar G. This enables agents to choose goals in contexts where planning is difficult. While formulating a useful grammar is non-trivial, methods for doing so are a productive area of current research [24, 25]. Program induction can also induce human-like biases, like preference for minimum description length or cache and reuse strategies [26, 27, 22]. Using symbolic formulation of goals also enables program reuse across domains—all while being more computationally efficient at the time of sampling a goal [28].

Last, we note that goal selection need not be a complex process. An agent could select goals at random, and then work towards achieving those goals. But without making an informed decision, achieving those goals may difficult or impossible.

3 Simulations

Aim. Program-based agents can select goals as informed by a grammar G, which reduces reliance on expensive planning. Here, we use the novel ShapeWorld task to compare a handcrafted, program-based agent to a planning agent. The grammar for the program-based agent captures the inherent symmetry in the transition function of ShapeWorld (e.g. `three(circle)`. We compare our programatic goal selection to the results from value iteration (the planning agent).

Task: Shapeworld. The agent first selects a single goal state that is different from previously selected goals, and is then presented with the initial shape configuration. The agent then traverses the state space to achieve the goal, ideally as efficiently as possible given the real-world cost of time. The agent traverses states by making shapes interact. Each shape is defined by three properties: number of sides (circle, triangle, square), level of shade (low, medium, high), and texture (plain, striped). Agents pick an actor shape (first click) and a recipient shape (second click) (Figure 2, left). Recipient shapes change by alternating texture and becoming closer to the

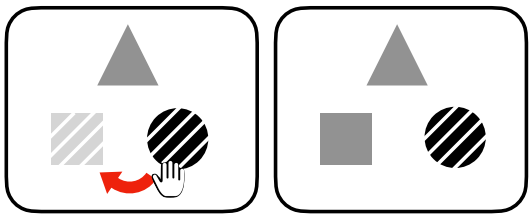

Figure 2: A ShapeWorld demo.

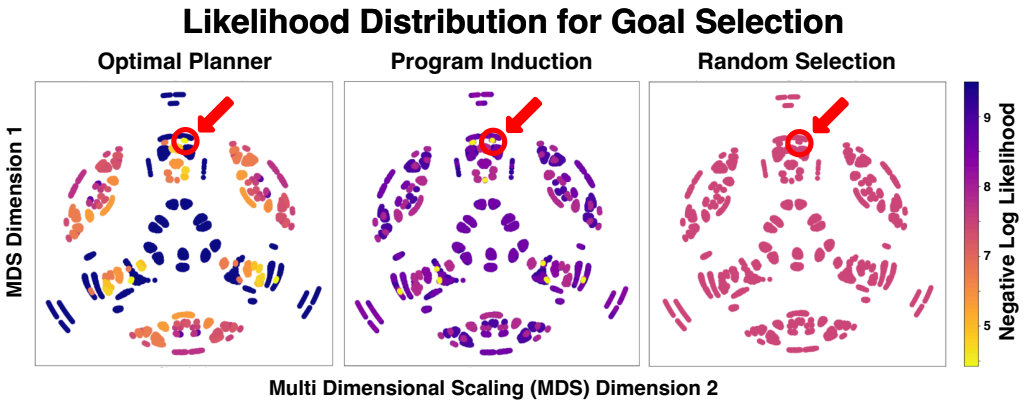

Figure 3: An embedding of the Shapeworld state space using multidimensional scaling according to the traversal distance between pairs of states. States are colored by negative likelihood (lower is better). The optimal agent shares the highest likelihood states (circled in red) with the program based agent. Outside of these few states, the agents diverge in their likelihoods.

shade of the actor shape. Recipient shapes also take on the number of sides of the actor shape with 0.8 probability. This yields $N = 5832$ states (shape configurations), which makes planning with a single goal state feasible (marginalizing over the state space) while making optimal goal selection difficult (solving the RL problem 5832 times, one for each goal state). Ultimately, this task is intended to be difficult yet manageable for human participants, making it suitable for testing human goal selection strategies.

Simulation. We designed a grammar (see appendix, Table 1) to prefer symmetric goal states. That is, states that have high rotational symmetry are easier to describe with few primitives. For example, the parsimonious program `two(circle)` symmetrically covers all states that have two shapes with the `circle` feature. Meanwhile, `two(square,(1,2))` is less parsimonious and covers similar, but not necessarily symmetrical states. To compare $P_G(s)$ and $P_{RL}(s)$, we simulated the likelihood distributions (Figure 3) for 1) an Optimal Planner using value iteration and Eq (2), 2) a Program Inductive Agent that uses Eq (5), and 3) a Random Agent as in Eq (6).

Results. Figure 3 uses multidimensional scaling to project the Shapeworld state space into two dimensions for visualization, with states colored by negative log likelihood (lower magnitude is better). The heat maps allow visualization of the divergent goal selection for each model. We note that both the Optimal Planner and Program Inductive Agent overlapped for most preferred goal states (e.g., those covered by `three(same)`). Meanwhile, the Program Inductive agent de-emphasizes non-symmetric states more than the Optimal Planner, as observed by a more uniform likelihood distribution across all the other states. The random agent is agnostic to the structure of the task, selecting goals according to a uniform distribution.

Discussion. We created an efficient goal-selecting agent that prefers symmetric goal states. Coincidentally, some symmetric states are also the easiest goal states to reach as indicated by value iteration (the Optimal Planner). This is because our transition function makes symmetric states easy to achieve. With a different transition function T, this grammar would likely fail to recommend easy-to-achieve goals. This is already seen in how the Program Induction Agent equally prefers non-symmetric goal states (assigns a flat likelihood distribution) in contrast to the Optimal Planner.

Overall, we demonstrate the feasibility of using program induction to propose candidate goals in a structured way. These goals may or may not align with high utility goals. Instead, program induction provides a promising, efficient, and structured method for solving the goal selection problem. What remains to be seen is (1) methodology for how to align program induction with prior knowledge about goal utility and (2) whether or not program induction describes human goal selection.

4 General Discussion

Program induction can embody domain-specific knowledge and inductive biases that can capture scenarios where people rely on their intuitions to set goals [22]. We demonstrate that program induction stands to provide a structured way of selecting goals when planning is too difficult. In our example, program induction can efficiently select symmetric goals. An added upshot of this model is that it captures how goal achievement can be intrinsically rewarding, since programs can map to a reward function [6]. This provides an example of how an agent can autonomously create intrinsic rewards for itself using inductive biases over selected goals.

Davidson et al (2024) demonstrated that this approach has limitations. It can be challenging to design a grammar G that perfectly aligns with the kind of goals people naturally come up with [21]. With recent progress of leveraging large language models as priors over programs [29], and methods that fine-tune a grammar to match human goal generation [21], there is great potential to empower this framework beyond simple hand-crafted grammars. That is, future directions can involve aligning program induction with existing knowledge about goal utility, and testing how well these methods describe human behavior.

Future directions of this framework can also involve approximating Bayesian inference methods as models for human goal switching. As grammar G usually covers an open-ended space, searching over this infinite space of possible programs has been notoriously intractable. But a rich literature of how people search over this space has begun to address this problem [24, 25]. Such algorithms could provide a computational account of how people set the next goal after achieving or failing the current one, how people realize that a goal is impossible to achieve, and when they have to change their mind as a result. Such directions can be used to investigate human behavior, potentially illuminating individual differences in goal selection strategies. As goal-centric learning has become an increasingly prolific area of research in human behavior [6], such directions provide potential for insight into how humans successfully solve difficult problems, or struggle to achieve their own goals.

References

[1] Edward L. Deci and Richard M. Ryan. *Intrinsic Motivation and Self-Determination in Human Behavior*. Springer US, Boston, MA, 1985.

[2] Nuttapong Chentanez, Andrew Barto, and Satinder Singh. Intrinsically Motivated Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

[3] Pierre-Yves Oudeyer and Frederic Kaplan. What is Intrinsic Motivation? A Typology of Computational Approaches. *Frontiers in Neurorobotics*, 1:6, November 2007.

[4] Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[5] Cédric Colas, Tristan Karch, Olivier Sigaud, and Pierre-Yves Oudeyer. Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: a Short Survey, July 2022. arXiv:2012.09830 [cs].

[6] Gaia Molinaro and Anne G.E. Collins. A goal-centric outlook on learning. *Trends in Cognitive Sciences*, 27(12):1150–1164, December 2023.

[7] Junyi Chu and Laura Schulz. "Because I want to": Valuing goals for their own sake. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 44(44), 2022.

[8] Richard Holton. Rational Resolve. *The Philosophical Review*, 113(4):507–535, 2004. Publisher: [Duke University Press, Philosophical Review].

[9] Michael Bratman. Intention, plans, and practical reason, 1987.

[10] Frederick Callaway, Bas van Opheusden, Sayan Gul, Priyam Das, Paul M. Krueger, Thomas L. Griffiths, and Falk Lieder. Rational use of cognitive resources in human planning. *Nature Human Behaviour*, 6(8):1112–1125, August 2022. Publisher: Nature Publishing Group.

[11] Olivier Sigaud, Gianluca Baldassarre, Cedric Colas, Stéphane Doncieux, Richard Duro, Nicolas Perrin-Gilbert, and Vieri-Giuliano Santucci. A definition of open-ended learning problems for goal-conditioned agents. *arXiv preprint arXiv:2311.00344*, 2023.

[12] Thomas M. Moerland, Joost Broekens, Aske Plaat, and Catholijn M. Jonker. Model-based Reinforcement Learning: A Survey, March 2022. arXiv:2006.16712 [cs, stat].

[13] David Abel, Mark K Ho, and Anna Harutyunyan. Three Dogmas of Reinforcement Learning. 2024.

[14] Richard S. Sutton. The reward hypothesis, 2004.

[15] Michael Bowling, John D. Martin, David Abel, and Will Dabney. Settling the Reward Hypothesis. In *Proceedings of the 40th International Conference on Machine Learning*, pages 3003–3020. PMLR, July 2023. ISSN: 2640-3498.

[16] David Abel, Will Dabney, Anna Harutyunyan, Mark K. Ho, Michael L. Littman, Doina Precup, and Satinder Singh. On the Expressivity of Markov Reward, January 2022. arXiv:2111.00876 [cs].

[17] Joar Max Viktor Skalse and Alessandro Abate. The Reward Hypothesis is False. September 2022.

[18] Keno Juechems and Christopher Summerfield. Where Does Value Come From? *Trends in Cognitive Sciences*, 23(10):836–850, October 2019.

[19] Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, August 1999.

[20] Tenenbaum Piantadosi, Rule. Learning as bayesian inference over programs. *Bayesian models of cognition: Reverse-engineering the mind*, 2024.

[21] Guy Davidson, Graham Todd, Julian Togelius, Todd M Gureckis, and Brenden M Lake. Goals as reward-producing programs. *arXiv preprint arXiv:2405.13242*, 2024.

[22] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[23] Noah D Goodman, Joshua B Tenenbaum, Jacob Feldman, and Thomas L Griffiths. A rational analysis of rule-based concept learning. *Cognitive Science*, 32(1):108–154, 2008.

[24] Neil R Bramley, Bonan Zhao, Tadeg Quillien, and Christopher G Lucas. Local search and the evolution of world models. *Topics in Cognitive Science*, 2023.

[25] Joshua S Rule, Steven T Piantadosi, Andrew Cropper, Kevin Ellis, Maxwell Nye, and Joshua B Tenenbaum. Symbolic metaprogram search improves learning efficiency and explains rule learning in humans. *Nature Communications*, 15(1):6847, 2024.

[26] Carlos G. Correa, Sophia Sanborn, Mark K. Ho, Frederick Callaway, Nathaniel D. Daw, and Thomas L. Griffiths. Exploring the hierarchical structure of human plans via program generation, 2023.

[27] Carlos G. Correa, Thomas L. Griffiths, and Nathaniel D. Daw. Program-based strategy induction for reinforcement learning, 2024.

[28] Bonan Zhao, Christopher G. Lucas, and Neil R. Bramley. A model of conceptual bootstrapping in human cognition. *Nature Human Behaviour*, 8(1):125–136, January 2024. Publisher: Nature Publishing Group.

[29] Lionel Wong, Gabriel Grand, Alexander K Lew, Noah D Goodman, Vikash K Mansinghka, Jacob Andreas, and Joshua B Tenenbaum. From word models to world models: Translating from natural language to the probabilistic language of thought. *arXiv preprint arXiv:2306.12672*, 2023.

A Appendix / supplemental material

Table 1: Probabilistic context free grammar for a program based agent.

Part		
Non-terminal Symbols	Production Rules	Description
S	and(S,S), S	Allow conjunction.
A	one(B), two(B), three(B), two(E), three(E)	Number of shapes considered.
A	two(F,E), three(F,E)	
A	one(B,C), two(B,D)	
A	two(E,D)	
B	square, circle, triangle	
B	low, medium, high	
B	striped, plain	
C	(0), (1), (2)	Single locations.
D	(0,1), (0,2), (1,2)	Diad locations.
E	same, different	Feature comparisons.
F	1, 2, 3	Number of features to compare.