Motivation and the Development of Activity

Tracking number: 370
machine learning, planning, cognitive modeling

Abstract

How do agents acquire competency in their environment? This is a core question of autonomous agents research. Competency may be hand-built into a system or it may be learned. We argue that for an agent to be truly autonomous, it must be capable to learn and direct its own behavior. In this paper, we present an architecture based on means-ends analysis planning that allows an agent to develop hierarchies of activity in a completely autonomous manner. Further, we show how an agent based on our architecture directs its behavior, creating its own goals using a technique called *planning to act*. The agent selects among these goals according to a motivational system, and in so doing, manages the tradeoff between exploring its environment and exploiting the knowledge it gathers. We present empirical results of such an agent working in simulated testbeds.

Introduction

When is an agent truly autonomous? Steels distinguishes *autonomous* systems from *automatic* systems in the following way:

> AI systems built using the classical approach are not autonomous, although they are automatic. Knowledge has been extracted from experts and put into the system explicitly [...] The resulting systems can solve an infinite set of problems [...] but these systems can never step outside the boundaries of what was forseen by the designers because they cannot change their own behavior in a fundamental way. (Steels 1995)

Related work by Luck & d'Inverno (1998), Norman & Long (1995), and Moffat & Fridja (1995) forward the ideas of *self-generation* (of goals) and *motivation* as a step towards autonomy by allowing an agent to choose which of its goals to purse dynamically under the control of its own motivational system. We contend that true autonomy, in the sense described by Steels, can only be achieved by a system which develops its own knowledge, and creates its own goals and behaviors grounded in this knowledge.

In this end, we have developed a theory of the development of activity in intelligent, autonomous agents that comprises three complementary processes. In the first process, an agent generates its own goals. Rather than choosing among arbitrary or exogenously-given states of the world, the agent generates goals of the form "engage in activity A" by a process we call *planning to act*. It then decides among available goals by ranking them according to a motivational system that takes into account the needs of the agent. In the second process, the agent uses a means-ends analysis (MEA) planner to build plans to achieve its goals. Plans that prove successful can be added to the agent's library of *activities* and retrieved for later use or revision. Finally, in the third process, our system learns from its experience to produce and refine operator models that inform the motivational system and planner. Guided by the motivational system, our agents learn new activities and their competence grows gradually to encompass all the actions the environment affords. Our approach parallels Piaget's developmental psychology of infants, in which executing and extending schemas (i.e., executing plans and extending them to achieve their preconditions) is rewarding (i.e., preferred by the motivational system) (Ginsberg & Opper 1988).

There are numerous alternative algorithms for solving pieces of this developmental puzzle. For example, reinforcement learning algorithms like Watkins' Q-learning (Watkins & Dayan 1992) learn approximations of how an agent's actions produce rewards in its environment. Q-learning and related reinforcement learning schemes have proven a successful approach to learning a variety of activities from robot control (Mahadevan & Connell 1992) to network packet routing (Boyan & Littman 1994). However, we prefer a developmental system based on MEA planning to model-free systems such as Q-learning for two reasons. First, means-ends analysis to plays an important role in human intelligence (Newell & Simon 1972). Second, means-ends analysis planning generates units of activity that are declarative and decomposable and requires the building of similarly declarative operator models that we believe can be useful in the development of other types of conceptual structure such as classes and language (Allen 1994).

In the section that follows, we offer a brief description of our planner and the modeling system that generates planning operators. Next, we motivate planning to act and introduce a motivational system that selects the goals of the agent. We show some results in simulated domains that suggest that the combination of planning to act, our motivational system, and an MEA planner manages to both explore the affordances of its environment and exploit opportunities for reward when it is advantageous.

Planning and Modeling

In our system, the development of activity boils down to the development of operator models that can be used by the planner to create useful behaviors. As operator models improve, so do the plans created by the planner.

We assume an agent is endowed with a set of primitive actions A, and a set of sensors S. A is assumed to be a set of discrete controllers, and S is a set of discrete or continuous sensors that change over time. When the agent executes an action *a*, it generates an *experience*, defined as a set of time series, one for each sensor in S, recorded as the action unfolds. The behavior of the sensors in each experience is determined by the action taken, and the state of the agent and its environment.

As an agent explores its environment it gathers experiences that fuel the operator modeling process. The first task of this modeling component is to classify experiences by their outcome. A single action may have many qualitatively different outcomes when taken in different situations. For example, a robot moving forward may bump into a wall, it may push an object, or it may have no interaction with another object of any kind. Each of these outcomes has its own consequence in planning, and each can be uniquely identified by how the agent's sensors change during the execution of the action. The modeling system must identify the qualitatively distinct outcomes of the agent's actions and build an operator model that represents each.

Our modeling system uses a clustering algorithm adapted to work with time series to partition experiences into operator models. Once experiences have been classified, the modeling system uses an induction algorithm to identify sensory states that are predictive of which outcome will unfold for a given action. These initial sensory states are called *initial conditions*, and correspond roughly to operator preconditions of classical planning.

As the operator models improve, so improve the plans generated by the planner. As plans improve, the goals of the agent are accomplished more readily, and thus, the agent's behavior can become more directed. Exactly how its behavior is directed, and thus, what type of activities the agent will develop, is the job of planning to act and the motivational system through the goals they produce.

Goal Selection

The process of goal selection determines the behavior of a developing agent. An effective goal selection scheme will produce goals which meet the following four criteria:

- **Plausibility**: goals should be possible for the agent to achieve.
- **Sustainability**: goals should attend to the basic needs of the agent. This includes vegetative needs like hunger, and fatigue, as well as needs like aversion to pain.
- **Learning**: goals should provide and agent with opportunities to learn about its environment.
- **Purposefulness**: goals should be generated with purpose. Let us define an agent that acts *purposefully* as one in which the decision to do something is always based on some criteria, and not simply random.

Managing to generate goals which meet these four criteria is not trivial. Generating plausible goals in a high-dimensional, continuous state space is a challenge on its own. Readers may recognize the management of the second and third criteria as the *exploitation versus exploration* tradeoff that has been well studied in the reinforcement learning literature (Thrun 1992; Dearden, Friedman, & Russell 1998): how does an agent split its time between trying to learn and trying to exploit the affordances of the environment for other benefit? An agent that spends too much time attending to learning will miss out on the benefits of activities it has already learned about, while an agent that spends too much time exploiting its known activities will miss out on undiscovered, and possibly more rewarding activities.

Goals in Development

The problem of how to generate goals for a classical planner is one that is traditionally not automated. Planning goals are, in general, specified as desirable sensory or perceptual states by some exogenous source, typically the experimenter. An account of development, though, must explain how an agent produces its own goals.

Generating goals automatically is, on the surface, a daunting task. If we take a reasonably complicated agent, such as the Pioneer-2 mobile robot, with some 65 real-valued sensors, the space of goals is multidimensional and unbounded. On top of that, many sensory states are not possible or highly improbable, such as positive translational velocity while stall sensors are reporting the robot is unable to move. Also, there is no obvious metric for preferring one sensory state over another. Is sonar-3 reporting a value of 715.3 a better goal that one in which sonar-3 is reporting 802.1, or are they equivalent for the purposes of the robot? On the surface, automatic goal generation for classical planning is highly under-constrained.

A simple observation on how human children spend their time during development provides leverage on the intractability of automated goal generation. Piaget noted that children seem to spend their time exercising *schemas*, or simple sensorimotor routines. What is rewarding to the child is the *activity*, not necessarily the perceptual state it brings about. Said differently, the goals of an agent, and those things that are rewarding to an agent, are *activities*, not sensory configurations. We call this philosophy of what an agent considers as possible goals *planning to act*, and this simple distinction is crucial to our theory of development for the following reasons:

- It constrains goal selection to a space of goals that is bounded and small. An agent looks at its list of things it has done, and picks the one it likes best to do again.
- It constrains goal selection to a space of goals that are physically possible. Rather than choosing arbitrarily

among sensory states, an agent selects from *things it has already done before*.

When our agent selects an activity to engage in, it selects an action *outcome* that it would like to reproduce. MEA planning becomes important when the outcome is not trivial to produce in the agent's current state.

All that remains of the specification of our goal generation system is to define a framework for preferring one activity over another that meets the criteria of sustainability, learning, and purposefulness.

A Model of Motivation

The planning to act framework reduces the goal generation process to a matter of evaluating the activities available to an agent and choosing one according to that evaluation. The details of how one does this determine how well the criteria of sustainability, learning, and purposefulness are attended to. Perhaps the simplest way to perform this evaluation is to define a preference relation, and simply choose the activity that is preferable to all others.

Let $\mathcal{A}$ denote the set of all the activities that an agent knows about. The preference relation for agent x, $\psi_x(a_1, a_2, s_t)$, holds iff in sensory state s_t agent x prefers to attempt activity a_1 over a_2. The goal of the system at time t, then, is a_g such that $\forall a_{n \neq g}(\psi_x(a_g, a_n, s_t))$.

If we were interested only in the learning criterion, we could base ψ on a metric such as information gain. Thus, every decision that agent x would make would be based on the amount of information to be gained about a particular action. Likewise, if the agent had some single, all-encompassing vegetative concern, such as keeping its battery from running out, we could base ψ on the expected loss or gain of battery power. This agent would constantly be attempting to maintain or increase its battery level. While both of these formulations of ψ satisfy the purposefulness criterion, both will also suffer the drawbacks of ignoring one of the other four criteria for effective goal selection. The exploratory agent will likely run out of battery power, while the agent obsessed with its batteries may never explore enough to find the most efficient policy for keeping its batteries running strong.

The relation ψ should reflect the variety of factors that motivate its behavior. For a mobile robot exploring the surface of Mars, for example, ψ may reflect three distinct factors: *fatigue*, or the need of the robot to recharge its battery when the voltage gets low, *crash avoidance*, or the need of the robot to keep from colliding with other objects at high speeds, and *curiosity*, or the need of an agent to improve its models of activity and the environment when they are not accurate enough. We call each of these components of behavior a *motivational factor*. Individually, each expresses a basic need of the agent, and collectively, they comprise behavior.

In our model of motivation, we represent each motivational factor F numerically with a *drive coefficient* $\mu_F(s_t)$. The drive coefficient expresses the relative importance of F in state s_t; the importance of the fatigue factor, for instance, may increase as the battery voltage decreases. Likewise, each activity a_n will have an expected motivational payoff $E_\Delta(a_n, s_t)$ when a_n is taken in state s_t, and the product of the coefficient and payoff yields the *factor value* of F for a_n. The sum of these products over $F \in \mathcal{F}$, where $\mathcal{F}$ is the set of all motivation factors, is the *desirability* of activity a_n in state s_t.

$$d(a_n, s_t) = \sum_{F \in \mathcal{F}} \mu_F(s_t) E_\Delta(a_n, s_t) \tag{1}$$

and

$$\psi(a_n, a_m, s_t) \iff (d(a_n, s_t) > d(a_m, s_t)) \tag{2}$$

It is worth noting that each factor is comprised of an internal (perhaps genetic) part, $\mu_F(s_t)$, and a part contributed by the environment which must be learned, the quantity $E_\Delta(a_n, s_t)$.

We have designed a simple motivational system based on three factors for the Pioneer-2 mobile robot. The coefficient function for the first factor, *fatigue*, depends on the robot's battery level, and is near zero when the battery is fully charged at 14V, indicating that fatigue plays no part in goal selection in the fully charged state. As the battery level drops, though, the coefficient rises, most dramatically in the range 10V-12V. Fatigue becomes an issue in this range, as the robot's effectors start to become unreliable when its voltage drops below around 10V.

We use a simulated *pain* sensor to implement a second motivational factor. Sudden contact with immovable objects produces surges of activity in the pain sensor. The coefficient μ_{pain} is always negative, indicating that the possibility of pain reduces the desirability of any activity with a potential for pain.

The final factor, *curiosity*, depends on the activity under consideration, and a measure called *accommodation*. Accommodation measures the agent's familiarity with a particular activity. If the activity is new, then accommodation is low, and curiosity makes a positive contribution to that activity's desirability. As the agent exercises the activity, and builds better models of how the activity works, accommodation increases, and the contribution of curiosity to the desirability of the activity tapers off. The curiosity coefficient reacts to the novelty of an activity according to a bell shaped curve, in a manner consistent with infant accommodation studies (Ruff & Rothbart 1996). Essentially, the curiosity factor asserts that all other things equal, an agent prefers to engage in activities it can learn most about.

How Curiosity Drives Development

The two primary jobs of the motivational system are to keep the agent out of trouble and to provide learning opportunities for the operator modeling process. Essentially, the motivational system must maintain the tradeoff between exploration and exploitation.

The combination of self-preservation motivational factors and the curiosity factor will manage this tradeoff in much the same way that counter-based directed exploration policies studied in the reinforcement learning literature do (Thrun

1992). That is, an agent has some preference towards improving deficiencies in known activity models over exploiting them for rewards.

In most reinforcement learning systems, an agent has access to the entire action and state space that does not change over its lifetime, which makes it possible to reason about areas of the state space it has not visited or actions it has not yet executed. For a system based on a planner with dynamically changing operator models (which define the state space used for planning), no such access can be guaranteed. Indeed, the space of activities it has to choose from comprise only those behaviors it has experienced. Such an agent cannot reason about an outcome it has not experienced (or has no reason to even believe the existence of), creating a potential problem. The motivational system clearly can manage to improve its existing models, but how such a system manages to learn *new* models may not be immediately evident.

We posit that useful exploration can result of exploitation done with incomplete or incorrect models. Said differently, a planner working with poor models will generate plans that create unexpected results from which the agent will learn new models, and our motivational system, coupled with the means-ends analysis planner and a mechanism for learning operators, is an example of this principle in action. In the following section, we will explore this hypothesis.

A Simple Learning Domain How exploration works in our system is best illustrated with an example. Consider a robot working in a parts preparation factory. The robot has a gripper and a paint gun, and a conveyor belt running in front of it, on which blocks pass by. The robot may attempt to pick up what is in front of it, open its gripper to drop what it is holding, fire its paint gun at its gripper, rest, or activate a refill switch to refill its paint gun.

The robot may sense whether it is holding something, whether that thing has been painted, its current fatigue, and its current paint level. The *outcome* of each of its five actions is based on the four observable sensors; if the robot sprays its paint gun, it may paint a block, it may repaint a block, it may paint its own gripper, or it may be out of paint and shoot nothing at all, for example. The robot gets feedback as to which outcome actually unfolds so that it may build an operator model to express the results.

This particular robot acts under the influence of 3 motivational factors: curiosity, fatigue, and *reward*, an exogenous signal which signals the robot when it has done something good (like paint and release a block) or bad (like attempt a refill when its paint gun is already full). Excessive fatigue may cause some actions to fail unexpectedly.

The robot's motivational system rates the known *outcomes* and geedily selects the the most desirable outcome as its goal. The MEA planner then generates a plan which will result in the desired outcome. The robot then executes the action, and the modeling component records the feedback and updates its models of each outcome's initial conditions and results.

So, exactly how does exploitation with poor models equal exploration? An example occurs almost immediately for our factory robot. Suppose the agent, right after being turned on, executes the GRASP command. It picks up a block, and receives feedback that it has experienced a new outcome, which for this discussion, we will call `pick-up`. Immediately, this outcome will be the most highly rated because of its novelty, and the goal selection system will suggest that the robot try and reproduce it. Due to its poor model of how `pick-up` works, the low-level control planner will suggest that activating the GRASP controller will produce the desired outcome. But, since the robot is already holding something, the GRASP controller experiences a different outcome, namely the `pu-fail` outcome, which comes about when the robot tries to grasp something when its gripper is already full. The robot has learned a new outcome due to planning with an incomplete model.

In the next section, we will see that examples of this type abound in this simple simulator, and that by about 60 invocations of the goal selection system, the robot has experienced all 19 of the outcomes of its 5 primitive actions. The governing principle is that planning to reproduce outcomes with flawed models produces flawed plans, which in turn produce unexpected results that are opportunities to generate new models or revise the flawed models.

Experiments

We have run two sets of experiments to show the behavior of an agent guided by our motivational system. The first is based on the block-painting robot, and the second is based on a general set of randomly generated Markov decision processes.

In both sets of experiments, we used a simple MEA planner and a fixed learning schedule. The planner is a simple generate-and-test planner that uses the simulator to evaluate whether or not it thinks a plan will succeed. It is purposely allowed to generate illegal plans if it has not experienced outcomes that would allow it to verify that the plans were indeed illegal. An example of this allowable illegal plan would be the GRASP plan described in the previous section. The agent does know about the `pick-up` outcome, and thus it can use it in plans, but since it has not experienced `pu-fail`, it cannot verify that the plan would fail. Once it has experienced the `pu-fail` outcome, the planner will be able to recognize that simply executing a grasp will fail to produce the `pick-up` result in this simple example.

Plots of desirability and outcome counts for the various activities of the factory robot domain are shown in figure 1. The results plotted are for a single run of the experiment, but results are similar across trials and with different starting configurations. By 60 steps into the simulation, all 19 outcomes have been experienced by the simulated agent. By about 100 steps into the simulation, the effects of novelty for 16 of the 19 outcomes have dropped to levels sufficient to make them undesirable except as steps in a plan to achieve some other outcome. Shortly after, the novelty of two of the remaining three (`repaint` and `drop-incomplete-block`) wears off, and `drop-complete-block` is clearly the most desirable outcome. At this point, there is no longer any incentive to explore, and the factory robot settles into an optimal strategy of executing plans that include alternately finishing blocks, resting, and refilling its paint gun. This is evidenced in the plot of outcome counts, in which only these three outcomes, and the required plan steps for achieving them continue to be executed.

Rather than hand-build increasingly sophisticated domains such as the block painting domain to show generality and scalability, we decided to apply our motivational system to randomly generated Markov decision processes. We used a simple algorithm to generate MDPs of a given size. The algorithm ensures that all states are reachable and that there are no dead ends in the MDP. Transitions between the states are considered "outcomes", and in our examples, 20% of all outcomes are assigned a positive or negative reward factor.

Figure 2 shows plots of desirability versus time and outcome counts versus time for a 5 action, 10 state, 22 outcome MDP. Outcome labels, which are randomly generated tokens like `A-SAC346-OC373`, have been omitted from the graph for clarity. As in the block painting domain, the 22 outcomes have all been discovered by step 100, and the most desirable outcome has distinguished itself by 200 steps into the simulation. Beyond that point, only six outcomes continue to be executed as plan components to exercising the outcome with the highest E_{Δ}^{reward}.

Figure 3 shows desirability and outcome counts versus time for a larger MDP with 10 actions, 20 states, and 44 outcomes. In this problem, our simple planner was bogged down with a very large search space of 44 operators, and due to time constraints, we cut the trials short after 210 steps. Still, the system managed to find 39 of the 44 possible outcomes after only 140 steps, and had become focused on a sequence of two rewarding outcomes shortly thereafter. Interestingly, as the graph of outcome counts shows, one of the highly rewarding outcomes had been discovered almost instantly. Over the course of exploring novel outcomes for the next 140 steps, the agent came across a new rewarding outcome. Soon after, the agent entered into a policy of alternatively planning for the original outcome and the new outcome, the two of which form a cycle, one leading into the other.

Conclusions

We have presented a theory of the development of activity in autonomous agents. This theory is based on means-ends analysis planning due to its developmental plausibility and the declarative, compositional nature of its representations, which we believe will useful in the development of other, related types of conceptual structure such as classes and language.

Our system comprises a cycle of three basic processes: goal selection, planning, and operator modeling. The difficult problem of autonomous goal selection for planning is addressed by *planning to act*, which states that the goals of an agent should be to reproduce past experiences, rather than reach arbitrary or exogenously provided sensory states. A motivational system then assigns *desirability* scores to the available goals which account for the vegetative and learning needs of the agent in order to define a preference relation ψ over the goals. ψ is then used to select the most desirable goal.

Empirical data in a simulated factory robot as well as randomly generated MDP domains indicate that the combination of our motivational system and a classical planner can effectively explore a domain, and once exploration is complete, effectively exploit the affordances of the domain.

References

Allen, J. F. 1994. *Natural Language Understanding*. Benjamin Cummings, 2 edition.

Boyan, J., and Littman, M. 1994. Packet routing in dynamically changing networks: A reinforcement learning approach. In Cowan, J.; Tesauro, G.; and Alspector, J., eds., *Advances in Neural Information Processing Systems 6*. Morgan Kaufmann Publishers, Inc.

Dearden, R.; Friedman, N.; and Russell, S. 1998. Bayesian q-learning. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*.

Ginsberg, H. P., and Opper, S. 1988. *Piaget's Theory of Intellectual Development*. Englewood Cliffs, NJ: Prentice Hall. 3rd Edition.

Luck, M., and d'Inverno, M. 1998. Motivated behaviour for goal adoption. In Zhang, and Lukose., eds., *Multi-Agent Systems: Theories, Languages and Applications - Proceedings of the Fourth Australian Workshop on Distributed Artificial Intelligence*, 58–73. Springer-Verlag: Heidelberg, Germany.

Mahadevan, S., and Connell, J. 1992. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence* 55(2–3):189–208.

Moffat, D. C., and Frijda, N. H. 1995. Where there's a will there's an agent. In *Intelligent Agents-Theories, Architectures, and Languages*. Springer Verlag. 245–260.

Newell, A., and Simon, H. 1972. *Human Problem Solving*. Prentice Hall.

Norman, T. J., and Long, D. 1995. Goal creation in motivated agents. In Wooldridge, M., and Jennings, N. R., eds., *Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890)*. Springer-Verlag: Heidelberg, Germany. 277–290.

Ruff, H., and Rothbart, M. 1996. *Attention in early development: Themes and variations*. New York: Oxford University Press.

Steels, L. 1995. When are robots intelligent autonomous agents? *Journal of Robotics and Autonomous Systems* 15:3–9.

Thrun, S. B. 1992. The role of exploration in learning control. In White, D. A., and Sofge, D. A., eds., *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*. Florence, Kentucky 41022: Van Nostrand Reinhold.

Watkins, C., and Dayan, P. 1992. Q-learning. In *Machine Learning 8*, 279–292.

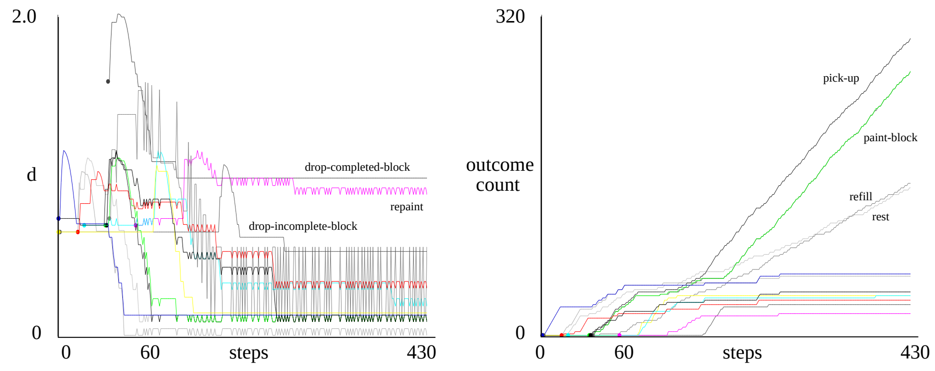

Figure 1: On the left, a plot of desirability levels versus time for each of the unqiue action outcomes in the block paining robot domain. On the right, a plot of the number of times each outcome has been observed over time.

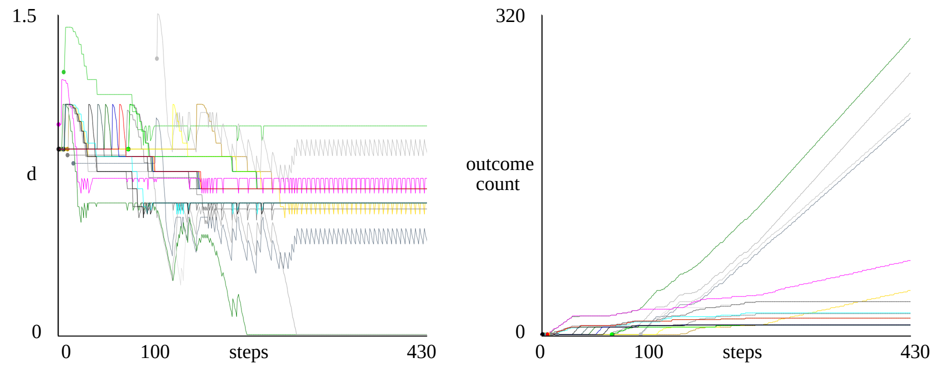

Figure 2: On the left, a plot of desirability levels versus time for the outcomes in a randomly generated Markov process domain with 5 actions, 22 outcomes, and 10 states. On the right, a plot of the number of times each outcome has been observed over time.

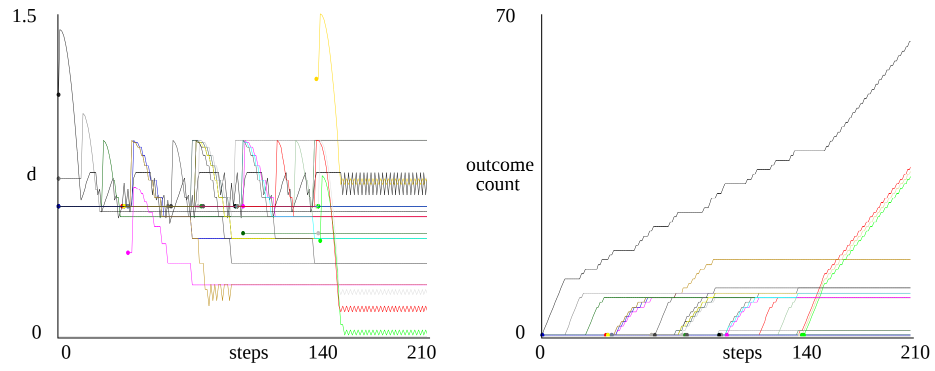

Figure 3: On the left, a plot of desirability levels versus time for the outcomes in a randomly generated Markov process domain with 10 actions, 44 outcomes, and 20 states. On the right, a plot of the number of times each outcome has been observed over time.