Deriving Behavior from Personality: A Reinforcement Learning Approach

Christopher Simpkins (chris.simpkins@gatech.edu)
Georgia Tech Research Institute, 250 14th Street, NW
Atlanta, GA 30318 USA

Charles L. Isbell, Jr. (isbell@cc.gatech.edu)
College of Computing, 801 Atlantic Drive
Atlanta, GA 80332 USA

Nicholas Marquez (nicholas.marquez@gatech.edu)
College of Computing, 801 Atlantic Drive
Atlanta, GA 80332 USA

Abstract

Creating artificial intelligent agents that are high-fidelity simulations of natural agents will require that behavioral scientists be able to write code themselves, not merely act as consultants with the ensuing knowledge acquisition bottleneck. We are designing a system that will make it possible to create rich agents using concepts familiar to behavioral scientists, such as personality models from psychology. However, translating personality models into the concrete behavior of an agent using currently available programming constructs would require a level of code complexity that would make the system inaccessible to behavioral scientists. What we need is a way to derive the concrete actions of an agent directly from psychological personality models. This paper describes a reinforcement learning approach to solving this problem in which we represent trait-theoretic personality models as reinforcement learning agents. We validate our approach by creating a virtual reconstruction of a psychology experiment using human subjects and showing that our virtual agents exhibit similar behavior patterns.

Keywords: Agents; Reinforcement Learning; Personality

Introduction

There is tremendous interest in creating synthetic agents that behave as closely as possible to natural (human) agents. Rich, interactive intelligent agents will advance the state of the art in training simulations, interactive games and narratives, and social science simulations. However, the programming systems for creating such rich synthetic agents are too complex, or rather too steeped in computational concepts, to be used directly by the behavioral scientists who are most knowledgeable in modeling natural agents. Engaging behavioral scientists more directly in the authoring of synthetic agents would go a long way towards improving the fidelity of synthetic agents.

Our goal is to create a programming language that a behavioral scientist can use to write agent programs using concepts familiar to behavioral scientists. This task is complicated by the fact that the most popular and best understood personality models from behavioral science do not lend themselves to direct translation into computer programs. Requiring a behavioral scientist to specify behaviors in the detail required in even the most cutting edge purpose-built programming language would plunge the would-be behavioral scientist agent programmer right into a morass of complex computational concepts that lie outside the expertise of most dedicated behavioral experts. To solve this problem we need a way to get from personality models to behaviors, to derive specific agent actions in an environment from a personality model without having to program the derivation in great detail.

In this paper, we describe a way to model motivational factors from trait-oriented personality theory by reinforcement learning components. We describe a virtual agent simulation that reconstructs a human subject experiment from psychology, namely some of Atkinson's original work in achievement motivation and test anxiety, and show that our simulation exhibits the same general behavior patterns as the human subjects in Atkinson's experiments. First, we briefly discuss relevant personality research and provide some background.

Personality

Personality is a branch of psychology that studies and characterizes the underlying commonalities and differences in human behavior. Within psychology, there are two broad categories of personality theories: processing theories, and dispositional, or trait theories. Social-cognitive and information-processing theories identify processes of encodings, expectancies, and goals in an attempt to characterize the mechanisms by which people process their perceptions, store conceptualizations, and how those processes drive their interactions with others (Dweck & Leggett, 1988; Cervone & Pervin, 2009; Cervone & Shoda, 1999). A strength of processing theories, especially from a computational perspective, is that they provide a detailed account of the cognitive processes that give rise to personality and drive behavior. This strength is also a drawback – processing theories tend to be detailed and often low-level (though not as low-level as cognitive architectures, which we will discuss below), and this makes them less intuitive and less suited to describing personality in broad, easily understood terms.

Trait theories (Cervone & Pervin, 2009), the most well-known example of which is the Five-Factor model (McCrae & Paul T. Costa, 2008), attempt to identify stable traits (sometimes called "trait adjectives") that can be measured on numerical scales and remain invariant across situations in determining behavior. A strength of the trait approach is that

they are well-suited to describing individuals in broad, intuitive terms. Two drawbacks of the approach are that there is not yet widespread agreement on a set of truly universal traits (or how many there are), and it is not clear how trait models drive behavior. A promising line of research by Elliot and Thrash (Elliot & Thrash, 2002) is working towards solving these problems by integrating motivation into personality in a general way. The work of Elliot and Thrash particularly supports the approach we present here, as they show that approach and avoidance motivation underpins all currently popular trait theories.

While debate continues about the merits and drawbacks of the different approaches to personality, the psychology community is also attempting to unify personality and motivation theory (Mischel & Shoda, 2008). While the work we present here is focused on bridging the gap between the descriptive power of trait-oriented models and the behavior that arise from them, we consider this work to be complementary to work in encoding information processing theories. In the future, rich computational agents may be built by combining approaches.

Reinforcement Learning

One can think of reinforcement learning (RL) as a machine learning approach to planning, that is, a way of finding a sequence of actions that achieves a goal. In RL, problems of decision-making by agents interacting with uncertain environments are usually modeled as Markov decision processes (MDPs). In the MDP framework, at each time step the agent senses the state of the environment and executes an action from the set of actions available to it in that state. The agent's action (and perhaps other uncontrolled external events) cause a stochastic change in the state of the environment. The agent receives a (possibly zero) scalar reward from the environment. The agent's goal is to find a *policy*; that is, to choose actions so as to maximize the expected sum of rewards over some time horizon. An optimal policy is a mapping from states to actions that maximizes the long-term expected reward. In short, a policy defines which action an agent should take in a given state to maximize its chances of reaching a goal.

Reinforcement learning is a large and active area of research, but the preceding is all the reader needs to understand the work presented here. More detail can be found in (Sutton & Barto, 1998; Kaelbling, Littman, & Moore, 1996).

Modeling Personality with Reinforcement Learning

The essential idea behind modeling personality traits with reinforcement learning is that each motivational factor can be represented by a reinforcement learning component. In psychology, the inherent desirability or attractiveness of a behavior or situation is referred to as *valence*. For a person high in success approach motivation, behaviors or situations that provide an "opportunity to excel" will have high valence, while other behaviors will have lower valence. The notion of valence translates fairly directly into the concept of reward in reinforcement learning. Just as people with certain motivational factors will be attracted to high-valence behaviors, a reinforcement learner is attracted to high-reward behaviors. This is the basis for modeling motivational factors with reinforcement learning components. By encoding the valence of certain behaviors as a reward structure, reinforcement learners can learn the behavioral patterns that are associated with particular motivational factors. This is a powerful idea, because it allows an agent author to write agent code using motivational factors while minimizing the need to encode the complex mechanisms by which such factors lead to concrete behavior.

A critical aspect of trait theory is that traits can have interactive effects. It is clear that a person who is high in achievement motivation will "go for it" when given the opportunity and that a person who is high in avoidance motivation will be more reserved. But what happens when a person is high in both motivations? Such interactive effects cannot be ignored in a credible treatment of personality, but it is hard to predict the behavioral patterns that will arise from given combinations of motivational factors. One can imagine the code complexity that might result from trying to model such interactive effects with production rules or other traditional programming constructs. As we demonstrate later, our reinforcement learning approach handles such interactive effects automatically.

It is important to note that we are not creating a new theory of personality. We are creating a computational means of translating existing theories of personality from *psychology* (not computer science) into actions executed by synthetic agents. We are also not committing to a particular theory from psychology, but rather supporting the general category of trait theories of personality which, until now, have not been directly realizable in computer agents.

In the remainder of this paper we discuss some related work in agent modeling, present our virtual reconstruction of a human subject experiment using our reinforcement learning approach, and discuss the promising results and their implications for future work.

Related Work

There is a great deal of work in modeling all sorts of phenomena in synthetic agents. Cognitive architectures provide computational models of many low-level cognitive processes, such as memory, perception, and conceptualization (Jones, 2005; Langley, Laird, & Rogers, 2008). Cognitive architectures support scientific research in cognitive psychology by providing runnable models of cognitive processes, support research in human-computer interaction with detailed user models (John, 1998), and can serve as the "brains" of agents in a variety of contexts. The most notable and actively developed cognitive architectures are Soar (Laird, 2008) and ACT-R (Anderson, Bothell, & Byrne, 2004). Recently, some effort has gone into integrating reinforcement learning into Soar (Nason & Laird, 2008). While RL is used to improve

the reasoning system in Soar, we are using RL to support new paradigms of computer programming for agent systems. In general, our work differs from and complements work in cognitive architectures in that we are drawing on psychological theory that is expressed at a much higher level of abstraction. Cognitive psychology and AI have often built on each other. Indeed, cognitive psychology is the basis of cognitive architectures in AI. Our work is an attempt to bring in mainstream personality psychology as a basis for building intelligent agents, which we hope will complement the detailed models of cognitive architectures in creating rich synthetic agents.

There is a large and rich body of work in believable agents. Mateas and Stern built on the work of the Oz project (Loyall & Bates, 1991) in creating a programming language and reactive–planning architecture for rich believable agents. They implemented their theory in the computer game Facade, a one-act interactive drama in which the player interacts with computer simulated characters that provide rich social interactivity (Mateas & Stern, 2004). Gratch, Marsella and colleagues have a large body of work in creating rich simulations of humans for training simulations that incorporate models of appraisal theory and emotion (Gratch & Marsella, 2005; Swartout et al., 2006). A distinctive feature of the work of both Mateas, et. al., and Gratch, et. al., is that they are dealing with the entire range of AI problems in creating believable agents that sense, act, understand and communicate in natural language, think, and exhibit human-like personalities. Our work differs from other work in personality modeling in that we are not attempting to simulate personality, but using definitions of personality to drive the behavior of synthetic agents. We want to derive behavior that is consistent with a given personality model, but not necessarily to ensure that the agent gives the appearance of having that personality.

Experiments

To test our claim that personality can be modeled by reinforcement learning components, we created a population of simple two-component multiple-goal reinforcement learning agents and ran them in a world that replicated experiments carried out with humans by psychologist John Atkinson. First we describe Atkinson's original research, and then discuss our virtual reconstruction of his experiments.

Atkinson's Ring Toss Experiment

John Atkinson was among the first researchers to study the existence and role of approach and avoidance motivation in human behavior. Prior to Atkinson's work, it was believed that test anxiety was equivalent to low achievement motivation. However, Atkinson showed that test anxiety is actually a separate avoidance motivation, a "fear of failure" dimension that works against and interacts with achievement motivation (Atkinson & Litwin, 1960). To test his hypothesis, he administered standard tests of achievement motivation and test anxiety to a group of undergraduate psychology students and devised a series of experiments which examined the effort put forth in achieving success in tasks such as taking a final exam. It is important to note that he did not measure the outcomes of the task, but rather the effort put forth in doing well in them. Thus, his experiments examined the relationship between motivation and *behavior*, not necessarily competence. One of his experiments, a ring toss game, produced results that clearly show the interplay of approach and avoidance motivation and is particularly well-suited to computer simulation.

In Atkinson's ring toss experiment, subjects played a ring toss game in which players attempted to toss a ring from a specified distance onto a peg. Subjects made 10 tosses from any distance they wished, from 1 through 15 feet, and the distance at which each subject made each toss was recorded. For analysis, subjects were divided into four groups according to their measures of achievement motivation and test anxiety so that the relationship between these motivations and their behavior could be analyzed. For each of the two measures – achievement motivation and test anxiety – subjects were classified as either high or low, with the dividing line between high and low set at the median scores in each measure. (For example, a H-L subject is high in achievement motivation and low in test anxiety). Subjects were divided into four groups – H-L, H-H, L-L, and L-H – and the percentage of shots taken at each distance by each group was recorded. We discuss his results and our simulation below.

Computational Models of Atkinson's Subjects

We reconstructed Atkinson's ring toss experiment in a computer simulation. We created 49 virtual agents that corresponded to each of the 49 human subjects in Atkinson's experiments, with the same distribution of high and low measures of achievement motivation and test anxiety. Simplified code for a representative student subject is presented in Figure 1. Since we did not have access to Atkinson's source data, we modeled high motivation measures as having a mean of 1.5 and low motivation with a mean of 0.5, both with standard Normal distributions (mean = 0, variance = 1) scaled by $\frac{1}{2}$, so virtual test subjects did not all have the same measures.

```
1 object Student((Achievement, 1.5 + X ~ N(0, 1) / 2),
2                (TestAnxiety, .5 + X ~ N(0, 1) / 2))
3 }
```

Figure 1: An agent representing a success-oriented student in Atkinson's ring toss experiment, containing two RL components representing high achievement motivation and low test anxiety. The code snippets presented here are simplified versions of the Scala code we used to run our experiments.

As discussed earlier, each of the motivational dimensions of the virtual subjects was implemented with reinforcement learning components that learned to satisfy the preference for perceived valence of behaviors (modeled as reward). For example, in the achievement motivation component (see Figure 2), the greater the distance from the peg, the greater the reward because it represents greater achievement. Similarly, in

the test anxiety component (see Figure 3), greater reward is given to closer distances, because they minimize, or "avoid" the chance of failure from a long-distance toss.

```
object Achievement extends AbstractRlComponent {

  world = RingTossWorld

  rewards = (1_foot_line -> 1,
             2_foot_line -> 2,
             // ...
             15_foot_line -> 15)

  actions = (play_1_foot_line,
             play_2_foot_line,
             // ...
             play_15_foot_line)
}
```

Figure 2: A reinforcement learning component representing achievement motivation.

```

object TestAnxiety extends AbstractRlComponent {

  world = RingTossWorld

  rewards = (1_foot_line -> 15,
             2_foot_line -> 4,
             // ...
             15_foot_line -> 1)

  actions = (play_1_foot_line,
             play_2_foot_line,
             // ...
             play_15_foot_line)
}
```

Figure 3: A reinforcement learning component representing Test Anxiety ('avoidance motive, a.k.a. "fear of failure"). Note that the rewards are inverted from the achievement motivation component, that is, the valence of avoiding achievement is higher.

Internally, each personality component is implemented with the standard Q-learning algorithm (Sutton & Barto, 1998). The ring toss world consists of 16 states – a start state and one state for each of the 15 distances, and 15 actions available in each state that represent playing (making a toss) from a particular distance. For readers interested in such details, each reinforcement learning component used a step-size parameter of $\alpha = 0.1$, a discount factor of $\gamma = 0.9$ (though discounting wasn't important given that the 15 states representing playing lines were terminal states, since each play was a training episode), and employed an ε-greedy action selection strategy with $\varepsilon = 0.2$. (Readers familiar with reinforcement learning will also notice that this game is roughly equivalent to a 15-armed bandit problem.) We emphasize that the details of the reinforcement learning algorithms are not essential to modeling motivational factors, and those details are hidden inside the implementation of the components. Indeed a major goal of our work is to simplify the task of writing synthetic agents by taking care of such details automatically.

Recall that reinforcement learning algorithms learn an action value for each action available in a given state. An action value for a state represents the expected total reward that can be achieved from a state by executing that action and transitioning to a successor state. For each of the components – Achievement and TestAnxiety – the action values represent the learned utility of the actions in serving the motivational tendencies the components represent. The Student agents take into account the preferences of the components – represented by action values – by summing their action values weighted by their component weights to get a composite action value for each action in a given state. If we denote each component's action value by $Q(s,a)$ and the weights by W, then the composite, or overall, action value is:

$$Q_{student}(s,a) = W_{Achievement} Q_{Achievement}(s,a) + \quad (1)$$

$$W_{TestAnxiety} Q_{TestAnxiety}(s,a) \quad (2)$$

For the virtual experiments, each component – Achievement and TestAnxiety – was run to convergence and then the student agents simulated 10 plays of the ring toss game, just as in Atkinson's experiment. We discuss the results of the experiment below.

Model Validation

A model is a set of explicit assumptions about how some system of interest works (Law, 2007). In psychology the system of interest is (usually) a human or group of humans. Our virtual reconstruction of Atkinson's experiments constitutes a computational representation of Atkinson's two-factor model of personality. Thus, our agents are simulation models of Atkinson's subjects (the students in his ring toss experiment). While the work presented here is only a proof of concept, we do hope to achieve a high level of validity as we refine our approach, so it will be useful to validate our models using techniques from simulation science (Law, 2007).

As we described earlier, Atkinson divided his subjects into four groups according to their measures (high or low) on achievement motivation and test anxiety. For each of these four groups – H-L, H-H, L-L, L-H – he recorded the percentage of shots that each group took from each of the 15 distances. We ran 10 replications of our simulation and recorded the mean percentages for each group and distance. For each percentage mean we calculated a 95% confidence interval. We consider a model to be valid if the confidence intervals calculated on the simulation percentage means contain the percentages obtained by Atkinson in his experiments with human subjects.

The validation results are presented in Table 1. Atkinson analyzed his experimental data by aggregating the shots taken by subjects into three "buckets" representing low, medium, and high difficulty. In Atkinson's analyses the dividing lines between the three buckets were set in four different ways with each yielding similar results. For brevity we present the division obtained by using both geographical distance and distribution of shots about the median shot of 9.8 ft, in other words, the dividing line one would choose by inspecting the histogram for distinct regions. This strategy resulted in the three buckets listed in the left column of Table 1. Each cell of the four subject groups – H-L, H-H, L-L, L-H - contains the

Table 1: Validation Results. For each subject group the percentage of shots taken by Atkinson's human subjects and by our simulation from each of three ranges is presented along with a 95% confidence interval for the mean percentage of shots in 10 simulated replications of Atkinson's experiment.

Achievement:	High	High	Low	Low
Test Anxiety:	Low	High	Low	High
	Atkinson	Atkinson	Atkinson	Atkinson
	Simulation	Simulation	Simulation	Simulation
Range	Conf. Int.	Conf. Int.	Conf. Int.	Conf. Int.
1-7	11	26	18	32
	7.7	14.0	5.6	8.5
	(4.0, 11.4)	(5.6, 22.4)	(1.4, 9.7)	(4.4, 12.5)
8-12	82	60	58	48
	75.4	69.0	74.4	80.0
	(65.1, 85.7)	(61.1, 76.9)	(62.0, 86.9)	(74.1, 85.9)
13-15	7	14	24	20
	16.9	17.0	20.0	11.5
	(8.8, 25.0)	(9.4, 24.6)	(8.3, 31.7)	(6.9, 16.2)

percentage of shots taken by Atkinson's subjects, the mean percentage obtained by running 10 replications of our simulation of Atkinson's experiment, and a 95% confidence interval for the mean percentage. While our model did not achieve formal validation, the general patterns of behavior are quite similar to Atkinson's human subject experiment and we consider these results to be a good proof of concept. We discuss some reasons behind these results and strategies for improvement below.

Discussion

We made several assumptions in our models that affected the validation results. First, because we did not have access to Atkinson's original data, only summary presentations, we did not know the exact distribution of motivational factors among his subjects, or even the scales used in his measures. We assumed normally distributed measures and tried several different scales before settling on the values used in the simulations reported here. Second, it is not clear how the valence of behaviors should be translated into reward structures for RL agents. We chose a simple linear reward structure in hopes that the system would be robust to naive encodings. To make our approach widely useful we will need to address the manner in which reward structures are determined. Third, we calculated aggregate action values by a simple weighted sum of component action values. We are currently investigating optimal arbitration of multiple RL components and hope to report results within the next six months.

We chose the Atkinson ring toss experiment on the advice of psychologists who recommended it as a well-known example of trait-oriented behavior theory, and because of its simplicity. However, our goal is to create large agent systems, so future work will need to address scalability – to greater numbers of trait factors and more complex worlds – and generalizability, or transferability, to other domains.

The algorithms we used also employed no optimization. Reinforcement learning suffers from the curse of dimensionality, and many techniques are being actively pursued to cope with the size of state spaces for realistic-size domains. Profitably employing reinforcement learning in agent programming systems will mean integrating scaling techniques such as function approximation (e.g., of action-value functions or state spaces) and decomposition techniques.

Finally, notice that the example code presented in this paper contains no logic for implementing behavior. The agents and the components are defined declaratively by specifying a state space, an action set, and a reward structure. The runtime system derives the concrete behavior of the agents automatically from these specifications. This technique, sometimes called partial programming (Simpkins, Bhat, & Isbell, 2008), is a key concept that increases the usability of agent programming by allowing programmers to specify *what* an agent is to do without getting mired in *how* the agent should do it.

Conclusions and Future Work

Reinforcement learning provides a promising approach to modeling personality traits and motivational factors in synthetic agents. In particular, it provides us with a means to create agent programming systems that are accessible to behavioral scientists and harness their knowledge directly while minimizing the need for complex programming. Much work remains to make this vision a reality, and our work is progressing on three paths. First, the integration of reinforcement learning into agent programming systems needs to be studied further so that we know when it is useful and how much detail

can be hidden from the agent programmer. Second, the examples presented here were written together so that the reward signals of each agent were directly comparable. If we want to enable large-scale agent programming, we must be able to arbitrate the reward signals of separately-authored reinforcement learning components (Bhat, Isbell, & Mateas, 2006). We are currently working on such an arbitration algorithm and hope to have results in the very near future. Finally, once the implications of integrating reinforcement learning components into agent models are sufficiently well understood and separately authored components can be combined in a modular fashion using an appropriate arbitration algorithm, we believe the best way to realize these benefits is in a language that incorporates these features in a coherent design. We are currently working on such a language, initially implemented as an internal domain-specific language (DSL) in Scala.

Acknowledgments

The authors are grateful for the support of the National Science Foundation and the Georgia Tech Research Institute. Patrick McNiel in the psychology department suggested the Atkinson example and was very generous in discussing our work and helping us understand personality psychology. Dr. Doug Bodner assisted us in applying validation techniques from simulation science.

References

Anderson, J. R., Bothell, D., & Byrne, M. D. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036—1060.

Atkinson, J. W., & Litwin, G. H. (1960). Achievement motive and test anxiety conceived as motive to approach success and motive to avoid failure. *Journal of Abnormal and Social Psychology*, *60*(1), 52–63.

Bhat, S., Isbell, C., & Mateas, M. (2006, July). On the difficulty of modular reinforcement learning for real-world partial programming. In *Proceedings of the twenty-first national conference on artificial intelligence (aaai-06).* Boston, MA, USA.

Cervone, D., & Pervin, L. A. (2009). *Personality: Theory and research*. John Wiley and Sons.

Cervone, D., & Shoda, Y. (1999). The coherence of personality: Social-cognitive bases of consistency, veriability, and organization. In D. Cervone & Y. Shoda (Eds.), (pp. 3–33). New York: Guilford Press.

Dweck, C. S., & Leggett, E. L. (1988). A social-cognitive approach to motivation and personality. *Psychological Review*, *95*(2), 256–273.

Elliot, A. J., & Thrash, T. M. (2002). Approach-avoidance motivation in personality: Approach and avoidance temperaments and goals. *Journal of Personality and Social Psychology*, *82*(5), 804–818.

Gratch, J., & Marsella, S. (2005). Lessons from emotion psychology for the design of lifelike characters. *Journal of Applied Artificial Intelligence (special issue on Educational Agents - Beyond Virtual Tutors)*, *19*(3-4), 215–233.

Ho, Y.-C., & Pepyne, D. L. (2001, December). Simple explanation of the no free lunch theorem of optimization. In *Proceedings of the 40th ieee conference on decision and control* (pp. 4409–4414). Orlando, Florida USA.

John, B. E. (1998, June). Cognitive modeling for human-computer interaction. In *Invited paper in the proceedings of graphics interface '98.* Vancouver, British Columbia, Canada.

Jones, R. M. (2005). An introduction to cognitive architectures for modeling and simulation. In *Proceedings of the interservice/industry training/simulation and education conference.*

Kaelbling, L. P., Littman, M. L., & Moore, A. P. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237-285.

Laird, J. E. (2008). Extending the soar cognitive architecture. In *Proceedings of the first conference on artificial general intelligence (agi-08).*

Langley, P., Laird, J. E., & Rogers, S. (2008). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*.

Law, A. M. (2007). *Simulation modeling and analysis* (4th ed.). McGraw-Hill.

Loyall, A. B., & Bates, J. (1991). *Hap: A reactive adaptive architecture for agents* (Tech. Rep. No. CMU-CS-91-147).

Mateas, M., & Stern, A. (2004). Life-like characters. tools, affective functions and applications. In H. Prendinger & M. Ishizuka (Eds.), (chap. A Behavior Language: Joint Action and Behavioral Idioms). Springer.

McCrae, R. R., & Paul T. Costa, J. (2008). Handbook of personality: Theory and research. In O. John, R. Robins, & L. Pervin (Eds.), (pp. 159–181). New York: Guilford.

Mischel, W., & Shoda, Y. (2008). Handbook of personality: Theory and research. In O. John, R. Robins, & L. Pervin (Eds.), (pp. 208 – 241). New York: Guilford.

Nason, S., & Laird, J. E. (2008). Soar-rl: Integrating reinforcement learning with soar. In *6th international conference on cognitive modeling.* Pittsburgh, PA.

Simpkins, C., Bhat, S., & Isbell, C. (2008, October). Towards adaptive programming: Integrating reinforcement learning into a programming language. In *Oopsla '08: Acm sigplan conference on object-oriented programming, systems, languages, and applications, onward! track.* Nashville, TN USA.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Swartout, W., Gratch, J., Hill, R., Hovy, E., Marsella, S., Rickel, J., et al. (2006). Toward virtual humans. *AI Magazine*, *27*(1).