Instinct Evolution in a Goal-Seeking Neural Network

Thomas E. Portegys

Illinois State University
Campus Box 5150
Normal, Illinois, USA 61790-5150
portegys@ilstu.edu

Abstract

Instincts are a vital part of the behavioral repertoire of organisms. Even humans rely heavily on these inborn mechanisms for survival. Many creatures, for example, build elaborate nests without ever learning through experience. This paper explores this evolutionary legacy in the context of an artificial goal-seeking neural network. An instinct is defined as a simple stimulus-response sequence that is triggered by environmental and other events. The well-known "Monkey and Bananas" problem is used as a task situation. Instincts are "hard-wired" neurons in the brain of a monkey. Using a genetic algorithm, a population of monkeys evolved to successfully solve the task that none were able to solve by experience alone. The solutions were also found to be quite adaptable to variations in the task; in fact more so than a hand-crafted solution.

Introduction

An instinct has been defined as "an innate tendency to action, or pattern of behavior, elicited by specific stimuli and fulfilling vital needs of an organism." (The Columbia Encyclopedia, Sixth Edition, 2006). All animals have instinctive drives (e.g. sex and aggression) and reflexes (e.g. blinking and gagging), but many simple animals also inherit instincts for complex behaviors such as courtship displays and nest-building. These animals are frequently either incapable of extensive learning through experience, such as insects, or occupy environmental niches in which it is most effective in terms of survival to inherit rather than learn these behaviors. Animals are often faced with situations in which experiential learning with its attendant mistakes is simply too expensive or too risky. For this reason there are also cases in which behaviors originally acquired through experience have become innate (Baldwin, 1896). Animals also do many things that do not have the immediate physiological "payoff" that obtaining food and water do, for example, but which are obviously beneficial for survival; the instinct to seek areas hidden from predators, for instance.

So it seems reasonable that simulating simple creatures should involve the use of instinctive behaviors. The importance of this was impressed on me after training a neural network to solve a maze (Portegys, 2005). Without a teacher, there would be little hope of success. Even with a teacher the incentives must be linked to existing needs, such as training an animal with food rewards. In a "natural" setting this can become artificial and contrived. Humans have innate behaviors to give and respond to social rewards (smiles, embraces, etc.), which constitutes a powerful mechanism for inculcating important behaviors, such as language acquisition. Social reinforcements can work for more complex creatures, but what about insects?

This paper investigates the evolution of instincts embodied in an artificial goal-seeking neural network. An instinct is defined as a simple stimulus-response sequence that is triggered by environmental and other events. It incorporates both drive and behavioral aspects of instincts. The well-known "Monkey and Bananas" problem is used as a task situation. This problem was first posed to study planning techniques, e.g. STRIPS (Fikes and Nilsson, 1971), and thus might seem a somewhat novel choice for neural network learning. Instincts are inborn "hard-wired" neurons in the neural network of a simulated monkey. A genetic algorithm is used to evolve a population of monkeys to solve the problem.

In this context, the evolution of instincts can be considered as an application of evolutionary computation, which has been applied to feedforward perceptron types of neural networks to construct network elements and weight their connections through genetic algorithmic approaches (Andriamasinoro, 2004; Igel and Sendoff, 2005; Weiß, 1994). These networks have been successful in such tasks as flocking (Baldassarre, *et al.*, 2003), foraging (Boshy and Ruppin, 2003), and cooperative nest building (Theraulaz and Bonabeau, 1995). In the Monkey and Bananas problem, a monkey must stack boxes in order to reach a goal of bananas, necessitating the evolution of control structures that achieve intermediate goals (box stacking and climbing) in order to obtain a final goal (bananas). The limited sensory apparatus of the monkey is such that it must retain some notion of the un-sensed state of the environment in order to succeed. I believe this state-retention is a unique feature of this project.

An additional purpose of this project is to further develop learning mechanisms suitable for a goal-seeking neural network called Mona. Although a connectionist architecture, Mona is more of a state-based planning system that a conventional pattern classifying neural network. It has exhibited complex behavior on a number of tasks, including cooperative nest-building (Portegys, 2001) (www.itk.ilstu.edu/faculty/portegys/programs/NestViewer/NestViewer.html). More recently (Portegys, 2005), it has learned mazes requiring retention of context information (www.itk.ilstu.edu/faculty/portegys/research/context-learning.html#simulator). A brief review of Mona follows.

A Review of Mona

This section describes the system that will incorporate the instinctive learning capability. Mona is based on the rationale that brains are goal-seeking entities (Bickhard, 1997). It has a simple interface with the environment, shown in Figure 1. All knowledge of the state of the environment is absorbed through senses. Responses are expressed to the environment with the goal of eliciting sensory inputs which are internally associated with the reduction of needs.

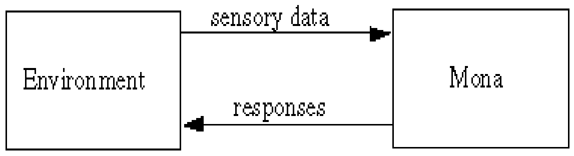

Figure 1: Mona/Environment Interface

Events can be drawn from sensors, responses, or the states of component neurons, calling for three types of neurons. Neurons attuned to sensors are receptors, those associated with responses are motors, and those mediating other neurons are mediators. Mediators can be structured in hierarchies representing environmental contexts. A mediator neuron controls the transmission of need through and the enablement of its component neurons.

To elucidate by example, consider this somewhat whimsical task: let Mona be a mouse that has been out foraging in a house and now wishes to return back to her mouse-hole in a certain room. For the sake of keeping peace with her fellow mice, she must not make the mistake of going into a hole in another room. Figure 2 shows her neural network at this juncture.

The triangle-shaped object at the bottom is the receptor neuron that fires once she has reached her hole; the inverted triangles are motor neurons that accomplish the responses of going to the correct room (Go Room), and going into the hole (Go Hole). The ellipses are mediator neurons. Each is linked up to a *cause* and *effect* event neuron. The "Hole Ready" mediator is not *enabled*, reflecting the importance of not going into a hole in the wrong room. The "Room Ready" mediator is enabled, signifying an expectation that if its cause event fires, its effect will also fire.

The "Home!" receptor neuron has a high goal value, indicating that it is associated with a need. Because of this, *motive* influence propagates into the network, flowing into motor neurons whose firings will navigate to the goal. Since the "Hole Ready" neuron is not enabled, the motive bypasses the "Go Hole" motor neuron in search of a mediator whose firing will enable "Go Hole". Since "Hole Ready" is an effect of "Room Ready", it flows into the "Go Room" motor via the enabled "Room Ready" mediator and causes it to fire (double outline).

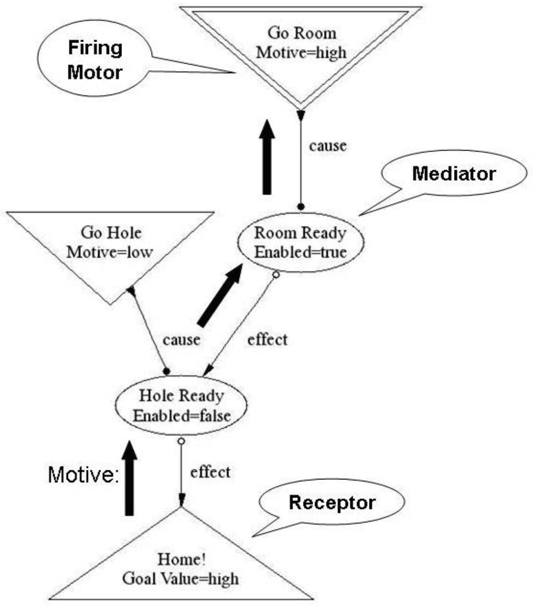

Figure 2: Initial Mouse Network

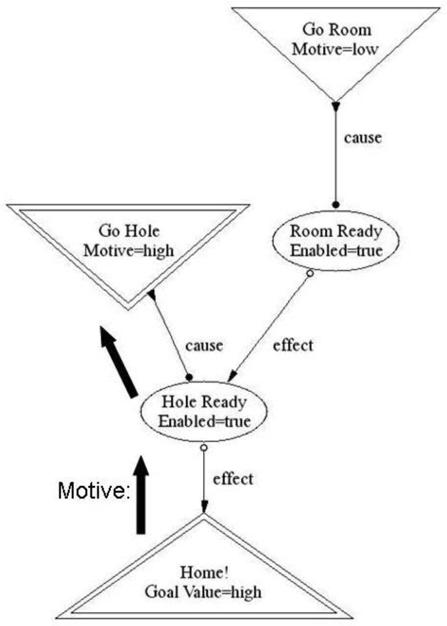

Figure 3: Final Mouse Network

The flow of motive illustrates how mediators representing contexts work together. The appropriate context for "Hole

Ready" is "Room Ready", which means that the latter should necessarily contribute something to the former in order to enable it. This something is called a wager. A wager temporarily modifies the enablement of a mediator that is the effect event of another mediator. It is called a wager because the base-level enablement of the wagering mediator will be evaluated based on subsequent firing of the effect neuron.

In Figure 3 the "Go Room" cause firing can be understood as a conditional probability event: given that Mona is in the correct room ("Room Ready"), she is quite certain that she can go into her own hole. This accomplished by a wager from "Room Ready", triggered by "Go Room", that boosts the enablement of "Hole Ready". After this enablement occurs, motive flows into the "Go Hole" motor neuron, causing it to fire. Subsequently the Mona senses that she is home in her hole.

Description

The Monkey and Bananas environment is shown in Figure 5. The monkey and three boxes are initially placed on the lower level of the floor. On the left upper level is a bunch of bananas. The environment is divided into discrete X and Y cells. The task is to gather and stack the boxes against the left wall in order to climb them to reach the bananas.

The monkey has three sensory capabilities: vision, direction, and box holding state. The vision sense allows the monkey to sense the state of the cell immediately adjacent to it in the direction it is looking. Possible values are: floor, wall, box, air, and bananas. The direction sense allows the monkey to determine which direction it is looking, left or right. The box holding state allows the monkey to determine if it is holding a box. The response capabilities are: go left, go right, climb, pickup, stack, and eat.

Instincts

An instinct is an entity that activates or inhibits a specific behavior by manipulating need and goal values. An instinct may activate periodically or in response to the activity of other instincts. An instinct is defined by several elements, as shown in Table 1. `

Event sequence	Stimulus-response sequence.
Need index	Identifies need.
Need value	Need increment when triggered.
Need frequency	Frequency of need triggering.
Need duration	Duration of need value.
Goal value	Need decrement when events fire.

Table 1: Instinct Definition

The event sequence is a either a 3 (S-R-S) or 5 (S-R-S-R-S) event sensory-response sequence. Longer sequences were not used to keep instinctive behavior simple. The event sequence is used to generate a mediator neuron and its component receptor and motor neurons. Instinctive mediators are unique in that, although they can be updated with respect to enablement, they cannot be deleted from the network. Normal mediators are subject to deletion when replaced by superior mediators.

The need index indicates which need is modified by the instinct, and the need value is the quantity of this modification. Need value may be either negative or positive; a positive need motivates behavior toward firing the associated mediator, and negative need establishes avoidance behavior. The need frequency indicates how often the instinctive need occurs; a value of 0 denotes a single occurence. Need duration indicates how long a need is expressed; a value of 0 denotes no time limit. Goal value indicates the amount of change to the need when the associated mediator neuron fires. Goal value may also be negative or positive. A positive goal value reduces the need, and a negative increases it. The latter can be used to trigger a subsequent behavior, and thus could be used to produce a chain of behaviors. An example of an instinct from a "successful" evolved monkey is shown in Figure 4.

```
events={
        stimuli=[Dont care,Look right,Dont care]
        response=Pickup
        stimuli=[Floor,Look right,Dont care]
        response=Go left
        stimuli=[Floor,Dont care,Hold]
}
needIndex=1
needValue=7.830277
needFrequency=9
needDuration=40
goalValue=1.873116
```

Figure 4: Example Instinct Values

The monkey instinct parameters are given in Table 2:

Maximum number of instincts	20
Minimum number of instincts	5
Maximum number of events	5
Minimum number of events	3
Number of needs	10
Maximum need value	10.0
Minimum need value	1.0
Maximum need frequency	10
Minimum need frequency	0
Maximum need duration	50
Minimum need duration	0
Maximum mediators	50

Table 2: Monkey Instinct Parameters

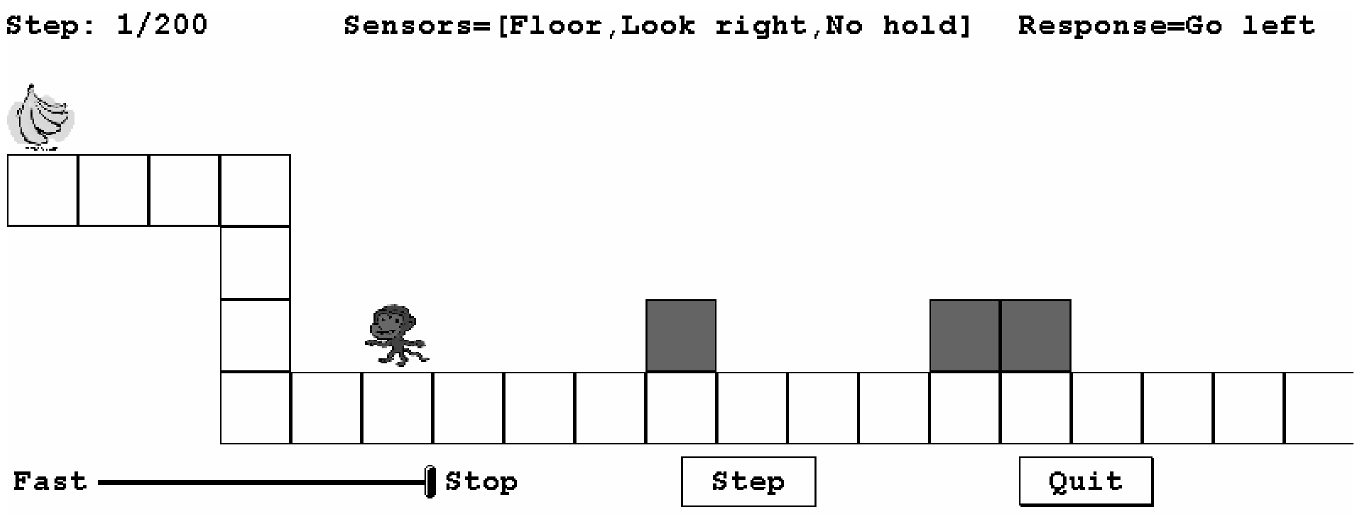

Figure 5: Graphical Interface

Evolution/Genetic Algorithm

An initial population of 20 monkeys was randomly generated using the parameters in Table 2. When a monkey was selected to run, its neural network was constructed using its instincts. In addition, each monkey was given an innate need for bananas, meaning the receptor sensing bananas was associated with a need and goal value unique to bananas.

Each monkey was given up to 200 steps to find the bananas. Fitness was calculated based on: finding the bananas, stacking boxes toward the wall, and speed (if bananas found). This rewarded monkeys who showed at least an ability to move boxes in the correct direction. A 100% fitness was assigned to monkeys finding the bananas. The next generation was created by selecting the fittest 10 monkeys, creating 8 new mutants and 2 new offspring. A mutant was derived from a random fit monkey by randomly replacing instincts with new random ones with a probability of 10%. Note that instincts themselves were not mutated. An offspring of fit monkeys contained a random selection of the parents' instincts.

Programming/Computing Environment

Mona is written is C++. The evolution programs include graphics written in Allegro (alleg.sourceforge.net), which is portable between the most popular OS/platforms, including Windows and various Unix/Linux machines. Two dual processor SUN Sparc machines were available for the evolution runs.

The open source C++ code, including some pre-built libraries, is available at:
www.itk.ilstu.edu/faculty/portegys/research.html#instinct

Results

As a base case, monkeys were run with no instincts, relying alone on experiential learning with a maximum possibility of 50 mediator neurons. Even when given many (> 1000) steps, none of the monkeys learned to find the bananas. Considering these monkeys started with no knowledge of the environment and limited sensory capabilities (only able to see adjacent cells), this was not surprising.

With the aim of segueing from simpler to more complex behavior, the first test evolved monkeys in a fixed environment, meaning that the boxes and the monkey started in the same locations for an entire evolution run, each of which consisted of 1000 generations. Since monkeys can possess both instinctive mediators and mediators created by experience, comparison runs were made to determine the influence of experiential learning. Specifically, all populations were given a maximum of 20 instinctive mediators, and populations also learning from experience could create an additional 50 mediators. An average of 10 runs is shown in Figure 6.

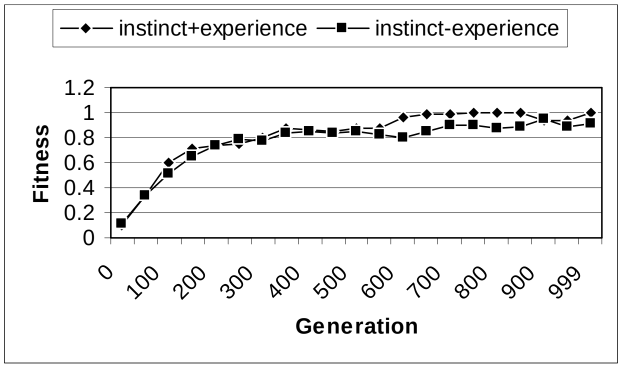

Figure 6: Fixed Environment Evolution

Data points on the graph represent averages of a population's fittest (selected) members in a 50-generation window of time. The graph shows that performance improves most rapidly in the initial 300 generations, gaining slowly after that. On inspection of the data, populations relying on instincts alone were able to achieve a 90-95% success rate on average. Adding experiential learning produces an improvement of about 5% above that, indicating that instinctive mediators were responsible for most of the fitness. In order to study the performance of instinct learning specifically, the remaining tests were made without experiential learning.

For the next test the environment was "scrambled", meaning that the placement of the boxes and the monkey varied randomly for each monkey test. So a monkey that solved a particular environment would likely be faced with a different environment in the next generation. The performance of populations under these conditions varied significantly, but the best were able to achieve a fitness of approximately 90%. Figure 7 shows the progress of one of the more successful populations.

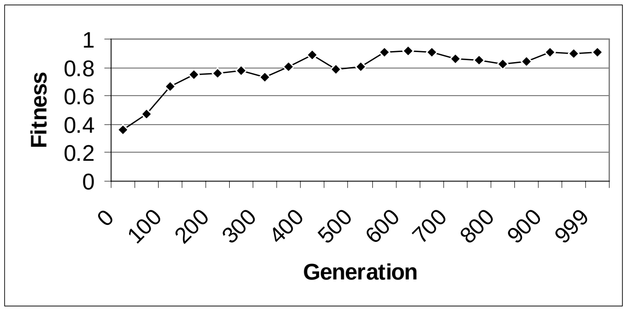

Figure 7: Initial Scrambled Environment

A consistent result was that the best populations of monkeys failed to achieve much better than 90% fitness. On closer inspection, some individual monkey fared better in some environments than others, so more often than not some monkeys in the population would be able to find the bananas, but individual monkeys did not have the ability to solve many varied environments. The instincts of such a monkey of course would have dominated the population.

This result inspired the next test, which was to take a population of monkeys evolved in a fixed environment and then expose them to scrambled ones. Interestingly, many such populations adapted relatively quickly (200 generations) to perform well in environments that they had never been exposed to and were able to perform at >95% fitness rate by the end of the run. However, as in the previous test, there were varied performances by populations. The evolution of one of the more successful populations is shown in Figure 8. Upon closer inspection, in these successful populations individual monkeys were more generalists, being able to solve many scrambled environments. Apparently for this task it is better to allow adaptation to a stable environment before varying it, rather than expect adaptation to an initially unstable environment.

A set of 10 evolved monkeys attempting to solve random Monkey and Bananas problems in real-time is available at itklinux.itk.ilstu.edu/~portegys/instinct-demo.html

A final comparison was made with a hand-crafted solution that was written to verify the feasibility of the task and to estimate the number of instincts that might be required, parameter ranges, etc. It was not a simple task to write, necessitating the 13 instincts shown in Appendix 1, but it performed perfectly in the fixed environment for which it was programmed. However, when exposed to scrambled environments, the average fitness dropped to less than 60%, meaning that the evolved instincts performed much better.

Observing a successful monkey at work, there is a great deal of seemingly erratic behavior, reminiscent of watching an ant crawl across a patch of ground, yet in the end the boxes are stacked, climbed, and the bananas obtained.

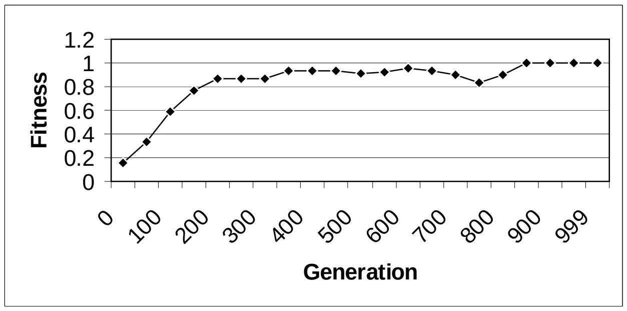

Figure 8: Fixed Followed by Scrambled Environment

Conclusions

The success of the instinct model suggests that it may be of further use in other problem environments where there is no teacher other than the environment itself, as is often the case with simple organisms. The fact that a set of brief behaviors could evolve to work together to solve a relatively complex task was quite remarkable, especially considering that the performance of evolved instincts was superior to a hand-crafted set of instincts. The instinct model has several novel features, such as time-based functions and the capability of some behaviors to motivate other behaviors, which could be useful in other applications.

It should be noted that a variation of the instinct model might be applicable to more conventional recurrent neural networks. Modules containing the time-based portions of instincts might be attached to the network as internal inputs, and the network trained to execute specific input/output sequences when appropriately activated.

Plans are underway to build a more complex environment involving predators and prey in which a combination of instinctive and experiential learning will allow organisms to survive through a balance of cooperative and competitive behaviors.

References

Andriamasinoro, F., 2004. Modeling natural motivations into hybrid artificial agents, Fourth International ICSC Symposium on Engineering of Intelligent Systems (EIS 2004).

Baldassarre, G., Nolfi, S. and Parisi, D., 2003. Evolving Mobile Robots Able to Display Collective Behaviors. Artificial Life, 9(3):255-268, Cambridge, MA, MIT Press.

Baldwin, M. J., 1896. A New Factor in Evolution. The American Naturalist, Vol. **30**, No. 354, 441-451.

Bickhard, M. H., 1997. Is Cognition an Autonomous Subsystem? In S. O'Nuallain, P. McKevitt, E. MacAogain (Eds.). Two Sciences of Mind. (115-131). John Benjamins.

Boshy, S. and Ruppin, E., 2003. Evolving Small Neurocontrollers with Self-Organized Compact Encoding. Artificial Life, 9(2):131-152, Cambridge, MA, MIT Press.

Fikes, R.E. and Nilsson, N.J., 1971. STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2(3-4), 189-208.

Igel, C. and Sendoff, B., 2005. Synergies between Evolutionary and Neural Computation. In M. Verleysen, editor, 13th European Symposium on Artificial Neural Networks (ESANN 2005), pp. 241-252.

Portegys, T., 2001. Goal-Seeking Behavior in a Connectionist Model, Artificial Intelligence Review, 16 (3):225-253.

Portegys, T., 2005. An Application of Context-Learning in a Goal-Seeking Neural Network, The IASTED International Conference on Computational Intelligence (CI 2005).

Theraulaz, G. and Bonabeau, E., 1995. Coordination in distributed building. Science, 269, 686-688.

Weiß, G., 1994. Neural Networks and Evolutionary Computation. Part I. Approaches in Artificial Intelligence. International Conference on Evolutionary Computation, pp. 268-272.

Appendix 1 – Hand-Crafted Instincts

```
// Look right for box.
  events={
    stimuli=[Floor,Dont care,No hold]
    response=Go right
    stimuli=[Box,Dont care,No hold]
  }
  needIndex=1
  needValue=4.000000
  needFrequency=0
  needDuration=50
  goalValue=0.200000

// Continue looking for box.
  events={
    stimuli=[Floor,Dont care,No hold]
    response=Go right
    stimuli=[Floor,Dont care,No hold]
    response=Go right
    stimuli=[Box,Dont care,No hold]
  }
  needIndex=1
  needValue=4.000000
  needFrequency=0
  needDuration=50
  goalValue=0.200000

// Pickup box.
  events={
    stimuli=[Box,Look right,No hold]
    response=Pickup
    stimuli=[Floor,Look right,Hold]
  }
  needIndex=2
  needValue=1.000000
  needFrequency=1
  needDuration=0
  goalValue=0.100000

// Turn left with box.
  events={
    stimuli=[Floor,Look right,Hold]
    response=Go left
    stimuli=[Floor,Look left,Hold]
  }
  needIndex=2
  needValue=1.000000
  needFrequency=1
  needDuration=0
  goalValue=0.100000

// Go left toward bananas.
  events={
    stimuli=[Dont care,Look left,Dont care]
    response=Go left
    stimuli=[Bananas,Look left,Dont care]
  }
  needIndex=2
  needValue=1.000000
  needFrequency=1
  needDuration=0
  goalValue=0.100000

// Continue going left.
  events={
    stimuli=[Dont care,Look left,Dont care]
    response=Go left
    stimuli=[Dont care,Look left,Dont care]
    response=Go left
    stimuli=[Bananas,Look left,Dont care]
  }
  needIndex=2
  needValue=1.000000
  needFrequency=1
  needDuration=0
  goalValue=0.100000

// Stack box on wall.
  events={
    stimuli=[Wall,Look left,Hold]
    response=Stack
    stimuli=[Box,Look left,No hold]
  }
  needIndex=3
  needValue=3.000000
  needFrequency=0
  needDuration=0
  goalValue=1.000000

// Stack box on box.
  events={
    stimuli=[Box,Look left,Hold]
    response=Stack
    stimuli=[Box,Look left,No hold]
  }
  needIndex=3
  needValue=3.000000
  needFrequency=0
  needDuration=0
  goalValue=1.000000

// Prevent stacking box out on floor.
  events={
    stimuli=[Floor,Dont care,Hold]
    response=Stack
    stimuli=[Box,Dont care,No hold]
  }
  needIndex=4
  needValue=0.000000
  needFrequency=1
  needDuration=0
  goalValue=-10.000000

// Prevent stacking box to right.
  events={
    stimuli=[Box,Look right,Hold]
    response=Stack
    stimuli=[Box,Look right,No hold]
  }
  needIndex=4
  needValue=0.000000
  needFrequency=1
  needDuration=0
  goalValue=-10.000000

// Prevent picking up stacked box
  events={
    stimuli=[Box,Look left,No hold]
    response=Pickup
    stimuli=[Floor,Look left,Hold]
  }
  needIndex=4
  needValue=0.000000
  needFrequency=1
  needDuration=0
  goalValue=-10.000000

// Climb two boxes.
  events={
    stimuli=[Box,Look left,No hold]
    response=Climb
    stimuli=[Dont care,Look left,No hold]
    response=Climb
    stimuli=[Bananas,Look left,No hold]
  }
  needIndex=5
  needValue=1.000000
  needFrequency=10
  needDuration=0
  goalValue=0.100000

// Climb wall and see floor on platform.
  events={
    stimuli=[Wall,Look left,No hold]
    response=Climb
    stimuli=[Bananas,Look left,No hold]
  }
  needIndex=5
  needValue=1.000000
  needFrequency=10
  needDuration=0
  goalValue=0.100000

// Eat bananas.
  events={
    stimuli=[Bananas,Look left,No hold]
    response=Eat
    stimuli=[Floor,Look left,No hold]
  }
  needIndex=0
  needValue=10.000000
  needFrequency=0
  needDuration=0
  goalValue=10.000000
```