Chapter 1

Exploration for Agents with Different Personalities in Unknown Environments

Sarjoun Doumit and Ali Minai
Complex Adaptive Systems Laboratory (CASL)
University of Cincinnati, Ohio, U.S.A.
sdoumit@ececs.uc.edu, aminai@ececs.uc.edu

We present in this paper a personality based architecture (PDA) that combines elements from the subsumption architecture and reinforcement learning to find alternate solutions for problems facing artificial agents exploring unknown environments. The underlying PDA algorithm is decomposed into layers according to the different (non-contiguous) stages that our agent passes in, which in turn are influenced by the sources of rewards present in the environment. The cumulative rewards collected by an agent, in addition to its internal composition serve as factors in shaping its personality. In missions where multiple agents are deployed, our solution-goal is to allow each of the agents develop its own distinct personality in order for the collective to reach a balanced society, which then can accumulate the largest possible amount of rewards for the agent and society as well. The architecture is tested in a simulated matrix world which embodies different types of positive rewards and negative rewards. Varying experiments are performed to compare the performance of our algorithm with other algorithms under the same environment conditions. The use of our architecture accelerates the overall adaptation of the agents to their environment and goals by allowing the emergence of an optimal society of agents with different personalities. We believe that our approach achieves much efficient results when compared to other more restrictive policy designs.

1.1 Introduction

Rodney Brooks' subsumption architecture(SA)[1] was meant as a reactive system for an exploratory robot that kept no memory or stored information from its environment. When applied in real life, the result was a robot that reacted to its surrounding's input and adjusted its actions accordingly, and while it did survive its environment, it did not store any meaningful abstraction of its experiences. Because the subsumption architecture (SA) decomposes a system into parallel tasks (or behaviors) of increasing layers of competence, (versus the typical functional decomposition), it allows the system to grow incrementally and renders it resilient

to sudden and unexpected changes in the environment. But on the other hand, the SA lacks the flexibility to implement higher-level concepts which allows it to be more useful for a wider scope of applications. For example, application involving software-based agents and robots with hybrid software/hardware controllers domains find it very difficult to apply a pure subsumption architecture to their design [2][3][4]. Our goal is to allow our agents to manipulate at a reflexive level, the knowledge that the environment presents and handle more complex and challenging missions by introducing hybrid learning algorithm (at a higher level. by introducing *personalities*). In our system, we consider the changing or driving forces that impacts the personality of the agent to be the activity of seeking different types of rewards from its environment and how this impacts the agent in chosing its best fitting personality. Using this paradigm, the design of our algorithm challenged us to address a fundamental issue in psychology, which is nature vs. nurture. When designing an agent, how much information should we embed into the agent and how much should we let the environment shape it on its own? We believe we achieved a balanced approach to solving this issue through our PDA architecture. In the next section we discuss our model, followed by the experiment environments and the results.

1.2 The PDA model

We present the schematic diagram in figure 1.1 of our PDA architecture by describing our *behavior*'s architecture, and then the personality's model.

1.2.1 The behavior architecture

We define a behavior to be a collection of *actions*. An *action* represents a procedural heuristic of several small processes that an agent can perform at a reflexive level. In relevance to Brooke's subsumption architecture, where we have different layers of behaviors, a behavior in our architecture can comprise multiple layers of actions. All actions receive input from the environment and forward their outputs to *junctions* via *connections*. A *junction* serves as an evaluator/relay for all its "inputs" and then outputs the resulting evaluation. In a simple behavior, shown in the diagram to the left in figure 1.1, we see a behavior comprised of 2 actions: `Action A` and `Action B` with a single junction `Junction J`, where the junction's output represents the overall behavior output. In more elaborate behaviors, we have layers of actions, such as the composite behavior to the right in figure 1.1. In these behaviors, the junctions evaluate the actions across the same level and also, from junctions that forward their outputs to other junctions, i.e. `Junction J` in level 1 subsumes/connects to `Junction J'` in layer 2 in `C-JJ'`. The final junction in the lowest layer handles the final output, or the final output of the behavior.

$$J = \sum_{i=1}^{n} W_i C_{i,J} + \sum_{k=1}^{m} W_k C_{k,J}$$

where J is the junction. i represents the available actions and k the number of other junctions that input into J. W_i is the weight of action i and $C_{i,J}$ is the weight of the connection from action i to junction J. (similarly for the junctions k). Every individual action and junction has an associated weight that determines the relevance of its output to the behavior's dynamics, i.e. `Action A` has weight `W-A`. Also, every connection has a weight that works as an attenuator or reinforcer to the output value it represents, i.e. the connection connecting `Action A` to `Junction J` is `C-AJ`. It is worth mentioning also that the layers are not hard boundaries, and that an `action A` in a high level can "subsume" (be connected via a junction to) the output of an `action A'` in a lower level. The behavior's architecture is based on a neural network design with more functionalities added at the neuron level.

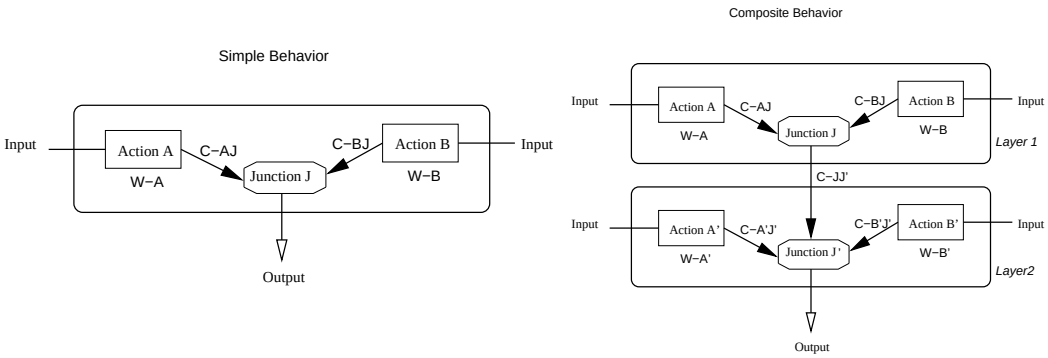

Figure 1.1: Behavior Architecture.

1.2.2 The personality's model

In our architecture an agent's *personality*, see figure 1.2, represents the collective values and weights of a behavior's components (actions, junction, connections). Our personality architecture applies a reinforcement learning architecture where the learning algorithm's objective is to maximize the available rewards present in the environment. This is achieved by predicting what best values for the weights and connections for a behavior to assume that would let the agent collect the largest number of rewards possible. As mentioned earlier, an action's weight determines the action's relevance and impact on the final output of the behavior, similarly connections' weights determine if this connection is valid, strong or non-existant. All these values have the effect of shaping the generic behavior architecture into more specialized behaviors, for example in the `Composite Behavior` to the right in figure 1.1, if the value of `C-BJ` $= 0$, then this means that `Action B`, and its contribution, are not part of the architecture. In practice, this change could mean the difference between an agent that has an *aggressive behavior* or a *careful behavior*. The ramifications of having different values for all the involved *parameters* is that we end up having an endless number of possible behaviors. This of course seems to be the natural way of the world, as in relevance to human beings, every person has a distinct personality that allows him/her to behave and respond uniquely to a situation. For simplicity and to facilitate the analysis of the system, we decided to make it a discrete learning problem where we define precisely what constitutes the different kinds of possible behaviors we wish to have. The personality therefore can pick from a *mood* collection (containing the lists of corresponding values and weights), the possible behavior type or *mood* it wishes to be in. The learning algorithm is divided into subtasks to maximize the corresponding reward components. For every type of identifiable reward, we assign a subtask that contributes towards maximizing the cumulative total reward. The PDA architecture is especially geared towards multiple reward sources and multiple goal requirements in order to bridge and marry these goals into a uniform behavior that maximizes the overall rewards. In the extreme case where there is one identifiable reward, the agent distinguishes between its *internal state of rewards* and the *external state of rewards*, (which is the internal state of rewards of other agents it comes into contact with), and assigns an *emotional maturity arbitrator* (or M) to each one of them that serves a predictor of the expected rewards. Let S_i, be the current behavior mood of the agent, where i represents its index in an array of available behaviors $[i..n]$ where n is some integer. Let R_i be a reward with index i associating behavior S_i to reward R_i. Also let λ_i be a decreasing factor (constant) for reward value R_i as time progresses and the agent is still in behavior S_i. Then we can write M as the expected sum of potential future rewards for time $t + n$ (where n

are discrete time units) as:

$$M_i(S_i) = E(R_{i,t+1} + \lambda_i R_{i,t+2} + \lambda_i R_{i,t+3} + ...\lambda_i R_{i,t+n})$$

$M_i(S_i)$ is the emotional maturity arbitrator value for behavior S_i at time t.

The winning mood (Si) is projected to the template behavior architecture to apply to the corresponding virtual behavior. In a way the template behavior architecture acts as a projector, and every behavior type is a projection or a virtual representation of the template behavior. (To clarify what we meant by stages in the abstract, the moods of the personality are actually the stages). Finally the rewards are abstractions for an accumulated stimulation or incentives. They are accumulated by the agent, and the agent can have either no, partial or full access to the rewards of other agents, depending on the personality of other agents (how much they like ot share). The information from other agents allows the agent to assess its performance and stance with respect to the rest.

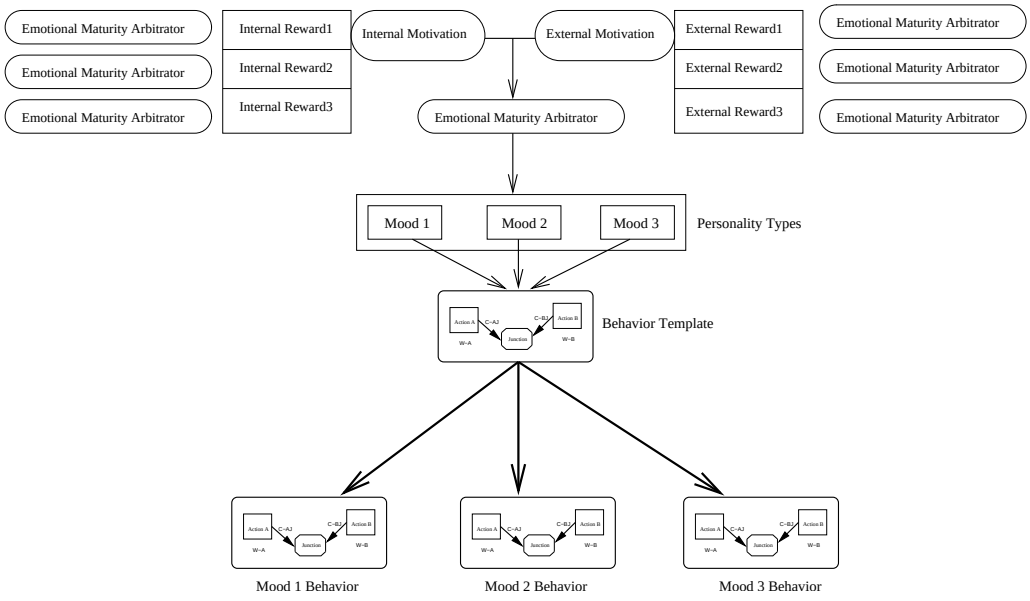

Figure 1.2: Personality Architecture.

1.3 Experiments

To test our propositions and validate the utility of our PDA, we developed a 3D simulated environment with artificial agents and applied our concepts to the agents in the terrain. We show how by applying the PDA architecture to agents with a specific goal allows the emergence of an optimal society of agents with personalities and behavioral dynamics fit to perform that task. For our testing, we chose the simplest and most visited example for this kind of applications, namely exploration by an autonomous agent. We chose this example in particular to i) compare the performance of our architecture and stem against a well understood and studied case, in which the subsumption architecture presents the ultimate solution given its simplicity and specification compatibility, ii) to show how the PDA performs in a situation where there is one main objective (i.e. exploration) which is against the PDA's strong points: i.e. joining

multiple objectives, iii) limited space to address more elaborate applications such as exploration and exploitation.

1.3.1 The environment

The environment(terrain) we're describing is a 3 dimensional lattice ($N \times N \times Z$) where (N) represents the discrete longitude and latitude value and (Z) represents the elevation value at every ($N \times N$) coordinate, or locations. The value of N ranges discretely from 0 still N. The elevation Z represents an associated cost matrix drawn over the 2 dimensional ($N \times N$) area (where the cost is a penalty value drawn from the agent's energy and time). The terrain is textured, to present to the agent another type of cost penalty, where every different type of texture represents a different resistance factor μ to the agent's velocity ($\overrightarrow{V}$) by $\mu \times \overrightarrow{V}$ and energy consumption.

1.3.2 The agents

Our simulated agents are relatively small robotic vehicles that are capable of limited data processing and communication capabilities. Each agent has a limited non-renewable energy source, and all its functionalities (mobility and communication) draw their energy requirements from it. Every agent is capable of detecting its own remaining energy level and can move in all directions of the compass ($0^o \rightarrow 360^o$) at a velocity that is determined by factors such as the terrain's texture and terrain elevation (uphill or downhill).

1.3.3 The behavior architecture

Our exploration behavior is made up of 5 different actions divided in 3 layers of subsumption dominance, see figure 1.3. In the top level we have the `Avoid Boundaries` and `Avoid Obstacles` actions, in the middle level we have the `Wander` action and in the low level we have the `Sweep` and `Line` actions. As each of their namesakes reveal, every action performs in manners that reflect that sub-behavior. The `Avoid Boundaries` action allows the agent to steer away areas to avoid crossing into territory that is not part of the mission. `Avoid Obstacles` on the other hand helps direct an agent around an obstacle (if possible), and allows it to resume afterwards whatever course it was taking, once it cleared the obstacle. `Wander` is an action for the agent to move randomly in the terrain, `Sweep` makes the agent systematically explore every location in a specified area, while `Line` makes the agent explore in a straight line. We also devised 3 types of moods that affect the behavior architecture's variables shown in figure 1.3 (i.e. the weights and connections). The moods or personality types are shown in figure 1.4 and are `Blind`, `Tolerant` and `Dedicated`. In the coming paragraph, the terms high, low, moderate and average represent the values that one might assume for the simulation. They are in reference to each other and do not represent a specific number. In the simulation results we will show the numerical values that we used. The `Blind` personality is usually characteristic of a fresh agent because it has no rewards of any kind. It does not "worry" much about negative rewards (hard terrain), for its `Avoid Obstacles` and `Avoid Boundaries` it has relatively low weights. Its `Wander` action is extremely random, and its `Sweep` areas are relatively large and its weights for `Sweep` are higher than that of the `Line`. (It appears as if it is in "desperate need" to accumulate rewards no matter what the cost). On the other hand,the `Dedicated` personality is usually of an agent that has usually accumulated one type of reward more than the other. Its weights for the `avoid boundaries` and `Avoid obstacles` is high, while its `Wander`'s direction values are within a short range from each other and its weight is low. Its `Sweep`'s areas are relatively smaller, so its `Line`'s values. The `Tolerant` personality is usually for an agent

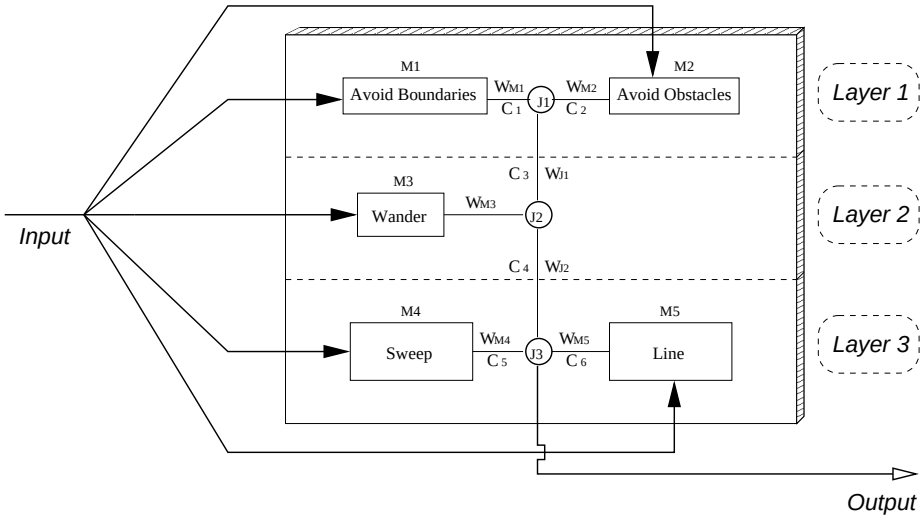

Figure 1.3: Exploration Behavior Architecture

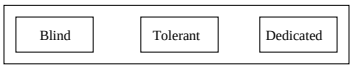

Figure 1.4: Personality Types (Moods)

that has a balance between its current values for its rewards. It has high weights for its avoid actions, its `Wander`'s direction values range moderately and it has a high weight for it, while it favors the `Line` action over the `Sweep` action for which it has moderate values (area and line). For the purposes of this experiment we made all personalities want to share all the information when nearby agents ask for it. We devised no negative reward for the communication, in order to keep the application simple and focused. Following are the results that we collected.

1.4 Results

For both our experiments our $N = 100$ and $Z = 0^o, 30^o, 60^o$. Agents move at a constant velocity V of 1 N unit per 1 time unit when at $Z = 0^o$ and texture $\mu = 1$. The agents have a power supply of 10 Joules, and their energy consumption matches that of small robots according to the average from various technical reports. The agents are distributed randomly on the terrain (not for the same experiment) and all start at the same time.

For our first experiment, we ran simulations to compared the average results of 5 agents that employ a pure subsumption architecture vs 5 agents that employ our PDA (with its 3 available moods). The agents using PDA all started with randomly selected personalities and the simulation ran for x time units. The results are shown in graphs 1.5 As the results show, the overall performance of the PDA is better than that of a pure subsumption architecture despite the simplicity of the application. Both architectures match up in the first phases of the experiment but as time progresses PDA's out performs subsumption.

For our second experiment shown in graph 1.6, we compared the average results of 5 agents that are always `Blind`, 5 agents that are always `Dedicated`, 5 that are `Tolerant` and 5 that

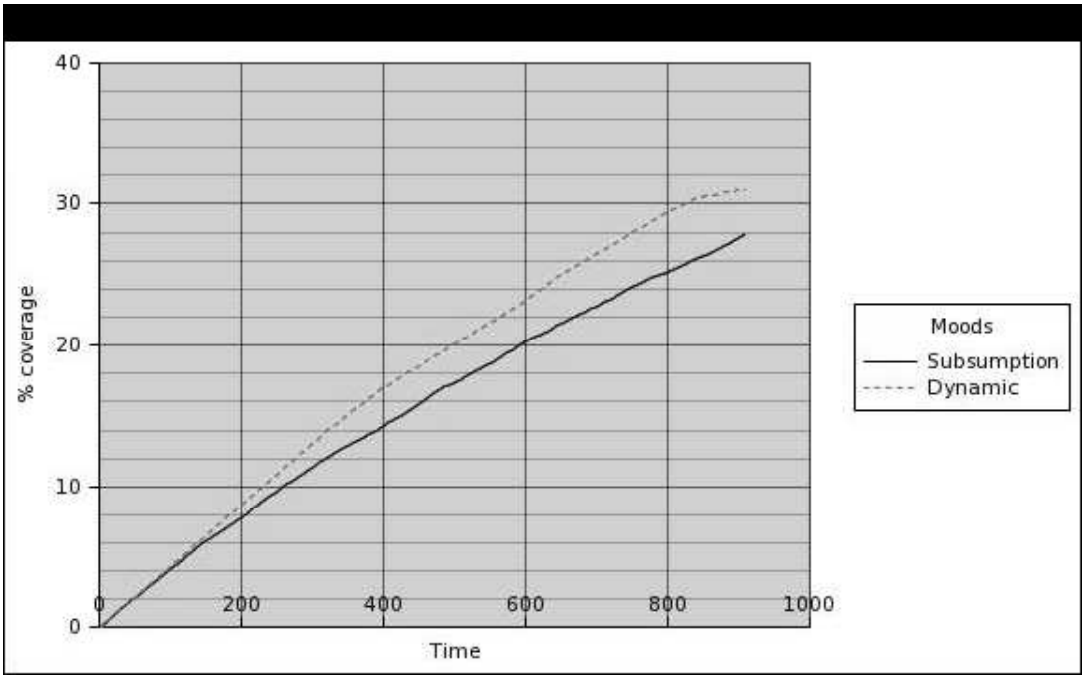

Figure 1.5: Exploration Coverage Ratio between PDA and Subsumption

have a dynamics personality. The purpose of this experiment was to compare (on average) the performance of these 4 personalities. The results showed an almost identical match amongst the `Dynamic` 31%, `Dedicated` 30.54% and `Tolerant` 30.3% with `Blind` 325.23% lagging behind. Due to the complexity of the experiments, the results can only show the emergent dynamics after applying our designs and architectures. Obviously our designation of the moods and the weights chosen played a pivotal role in deciding the outcome of the experiment, which brought us back to the question we asked before, how much information should the agent have before going to the world. From the results it does show that having a changing personality offers a slight advantage over static personalities, but then again the task at hand was not complex enough to allow the PDA to improve the general performance. For future work, this issue will be addressed and more in depth applications will be chosen for the test beds. As a conclusion, we believe that our architecture offers a first step and a work-bench in the right direction to improve upon the performance of autonomous agents with multiple goals in unknown environments.

Bibliography

[1] Brooks, Allen, "A robust layered control system for a mobile robot" (1986).

[2] Franklin, Stan, and Art Graesse, "Is it an agent, or just a program: A taxonomy for autonomous agents", *Tech. Rep. no.*, Institute for Intelligent Systems. University of Memphis, (1996).

[3] Rosenblatt, J., S. Willams, and H. Durrant-Whyte, "Behavior-based control for autonomous underwater exploration" (2000).

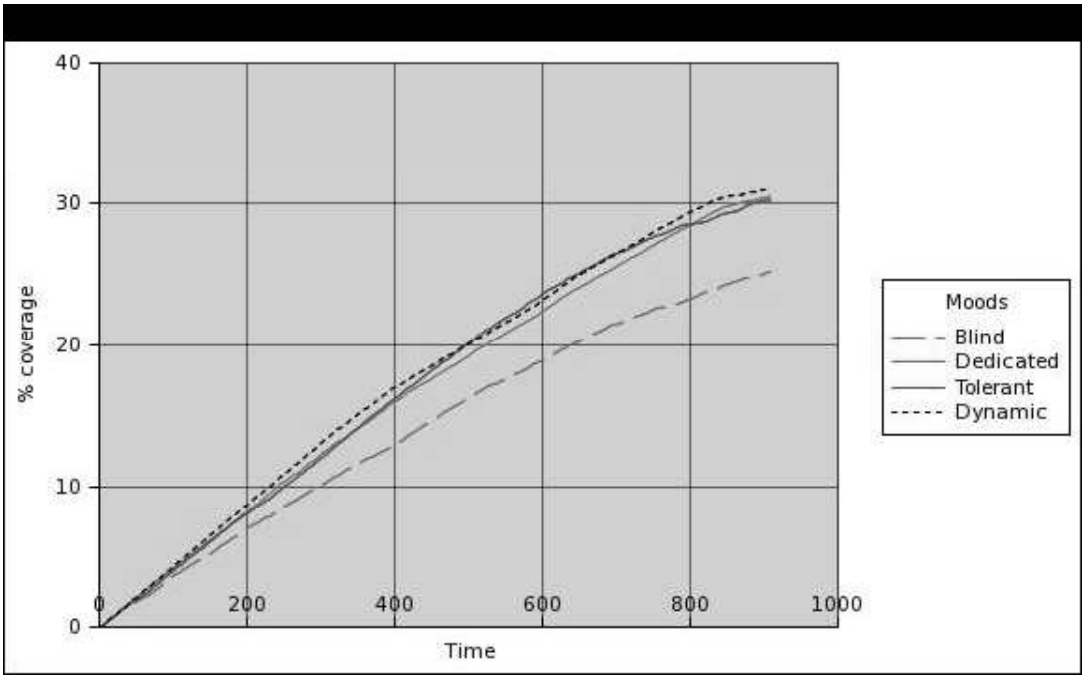

Figure 1.6: Exploration Coverage Ratio between different static and dynamic personalities

[4] University, North Carolina State, "Pipe-crawling robots designed to find earthquake, bomb survivors", *Tech. Rep. no.*, North Carolina State University, (1999).