Vrije Universiteit, Amsterdam
Faculteit der Exacte Wetenschappen

Master of Science Thesis

Target Selection and Multi-Objective Controllers in NEW TIES

by

Selmar Smit

Supervisors:
Robert Griffioen and Guszti Eiben

Second reader:
Martijn Schut

Amsterdam, 2007

Contents

Acknowledgements

In the first place, I would like to thank Robert Griffioen for his help and guidance during my graduation period. Our discussions and brainstorms were a great inspiration for this thesis. Furthermore I would like to thank Guszti Eiben for the opportunity to work on the NEWTIES project and his help revising this thesis.

I would like to thank Joeri Bekker and Evert Haasdijk for the great time and discussions we had. Finally I would like to thank Christian Tzolov for this technical assistance that enabled me to finish my experiments.

Selmar Smit
Amsterdam, July 2007

Remark

The experiments that are presented in this thesis are only the crown-jewels of the executed experiments. Most presented experiments needed excessive tuning and adaptation of the environment and the agent controller. Only a fraction of these experiments were successful and are presented in this thesis. An overview of the executed experiments can be found in Appendix E. The experiments that are performed for the poster 'Balancing quality and quantity in evolving agent systems' [1] are not listed in this overview.

Furthermore, much effort is put into debugging the current NEW TIES environment, fixing errors in the agent controller itself and creating data-analysis tools to be able to analyse the results.

This thesis presents therefore only a small portion of the work performed during the internship. This is chosen deliberately to be able to emphasize on the findings.

Executive Summary

In this thesis, we introduce some new methods that increase the power of NEW TIES agents with respect to decision making. The NEW TIES environment is an emulated world where a population of agents behaves in an a-life-like manner. Agents can navigate through this world by executing motor actions. Furthermore, they can pick up and eat objects they are standing on and agents can perform mate actions. If both parents decide to mate, a child is born. The controller of the child is created by combining the controllers of both parents. The challenges agents face are demanding and agents have to adapt in order to survive. In NEW TIES, survival selection is the enabler for evolution.

The first mechanisms that are proposed in this thesis are used for Target Selection. Target selection is the mechanism that supplies a target, when a certain action needs one. For example, when executing a pickup action, the controller has to supply which object have to be picked up. The choice of the target is essential for survival, some objects are to heavy to be picked up, others are to far away. In the proposed experiments, there are also poisonous objects, which can be lethal. We propose two different mechanisms for target selection and illustrate their advantages and disadvantages.

The results show that the speed in which a certain task does not significantly differ in both mechanisms, but the agents that were controlled by a Decision Tree with Attention Oriented Tree Traversal (AOTT) outperformed the agents with ordinary tree traversal on efficiency executing this task. Although the results show that the performance of agents with AOTT is more sensitive, it would probably lead to the highest performance.

Secondly, we address a mechanism that enables agents to deal with environments in which agents not only have to eat in order to survive, but they face other objectives too. We illustrate how the current controller, the Decision Q-Tree, which is purely reactive fails on these kind of environments. Furthermore, we give an overview of the currently known mechanisms and architectures that can deal with these multi-objective worlds. Most of these mechanisms are depended on an explicit fitness function, while the NEW TIES environment lacks such a function. Therefore, we propose a mechanism that is based on currently used architectures and works without a fitness function.

The results show that this mechanism, Behaviour Oriented Decision Tree (BODT), is capable of facing problems in multi-objective environments, while ordinary reactive Decision Trees fail. BODT-agents did not only evolve a controller that was capable of solving these problems, but also passed down this knowledge for many generations.

Finally, we discuss the effects of co-evolution and how agents can adapt their behaviour without being restricted to the ideas of the researcher.

Chapter 1

Introduction

1.1 General Introduction

Most people think that robots are a threat to us all, they defeat us on almost every task we can think of. They can build a car more precise than we can, our chess world-champion is no match for them and they can remember a lot more digits of Pi. Although some of these tasks are very challenging, robots still struggle with the most basic tasks of living; decision making. They are just "idiot savants", like Blumberg [2] called them, they have the knowledge to do the greatest things, but fail on the most simple tasks.

Most married man of this world know the problem, the taxi is waiting for you, but your wife is still not dressed. Although her wardrobe is twice as big as yours, she hasn't got anything to wear. Her problem is not what to wear, but what to wear not. There are just to many choices. What if people fail on decision making every day, on every issue? What if you cannot decide on what to do? What to eat? Or who to share your bed with? If we fail on decision making, we would do nothing, eat nothing and would never reproduce; Decision making is the key of living.

In this thesis we describe the world of NEW TIES*, a world full of agents struggling to survive due to their lack of decision making capabilities.

*New and Emerging World models Through Individual, Evolutionary and Social learning (NEW TIES), EU FP6 Project, `http://www.new-ties.org`

1.2 NEW TIES

Our investigations are carried out in the NEW TIES system. NEW TIES is a population-based adaptive system. It emulates a world where a population of agents behaves in an a-life-like manner. Agents can adapt to their environment through evolution and learning, where learning can be either individual or social [3]. These mechanisms enable them to face the challenges that researchers add to the environment. Worlds with rare of poisonous food increase the difficulty of survival, while trading opportunities can can enhance their life.

1.2.1 The NEW TIES Environment

With this system we can set up different types of worlds and different types of agents. Generally, a NEW TIES world is carried by space, time and energy. The world is a rectangular grid, time shifts by atomic time steps and energy changes in basic units. The laws of nature determine the behaviour of the world's inhabitants like agents and plants. Most importantly, it determines what agents can do in their environment, how much it cost in terms of energy and it determines plant growth. The latter taking care of energy production. The predator-prey relation between agents and plants changes the agent and plant distribution over the world sometimes leading to the extinction of one or both species.

From the agent's perspective, this is making NEW TIES a true survival game where the only objective is survival of the individual and agent population. To survive, agents have to solve at least one the challenges researchers have placed in the world, but accompanying this challenge there are always the basic survival tasks of finding food, mates, etc. Within the rules of the laws of nature NEW TIES agents solve this 'game' by means of adaptation mechanisms that can change their behaviour. Because of the complexity of the system unexpected agent behaviours can emerge solving this game.

1.3 Problem Statement

Target Selection
The behaviour of agents within the NEW TIES system is determined by a controller unit that decides which action should be performed and which object should be the subject of this action. For example, the controller could determine that the agent performs a pickup action, if it's standing on a plant, but could also decide to walk away from it. It is clear that determining the correct actions is essential for survival. If the controller never decides to eat, than that particular agent probably dies soon. But even if the controller manages to select the right action at the right time, success is not guaranteed. Agents can insist on mating with plants, or trying to pickup something that is out of their reach.

These are problems that most people consider as minor problems. A human being would never try to mate with an apple or pickup something that we cannot reach, but they are not. We are choosing actions and subjects belonging to these all the time. If we are

hungry we just get some food. But why do we choose that particular piece of food? Is it because we like that kind of food, or is it because it is just within reach and we do not want to get out of our chair in order to get something else. We use our own *target selection* mechanism to deal with these kind of choices and so should agents.

Multi-Objective Controllers

But these are not the only problems our agents face, there are more problems that are not evident at first sight, but can cause huge problems when simulating real environments. One of these problems arises when agents face a *multi-objective* world. for example, they have to stay alive and also perform some other task. This can easily be illustrated by the following case:

A certain agent that has the objective to run as far as possible, while staying alive, is using a decision tree (Figure 1.1) to determine what to do.

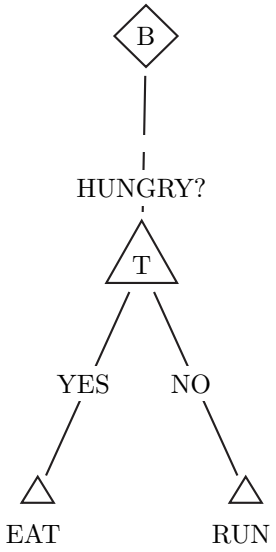


Figure 1.1: Example: Running Agent Decision Tree

This seems a useful tree that creates a very efficient behaviour. However, what happens is the following: The agent starts running until it gets hungry. It will sit down and start eating the sandwich it was carrying. After only one bite the hunger disappears. This is not because it was a very healthy sandwich, but because the agents energy just fell below its hunger threshold. One bite increases its energy to a level just above this threshold. So it decides to start running again. Within a few steps the agent is hungry again and sits down. This behaviour is repeated over and over again. This is certainly not the expected behaviour, because every human being would eat until its stomach is full and then starts running until it gets very hungry. This 'human' behaviour is far more effective, but although it looks quite simple, implementing it within a decision tree is not straightforward. The currently used standard decision tree in NEWTIES can only represent purely reactive behaviour, so to enable the agent to effectively choose its actions, some structural changes are needed.

1.4 Objectives

The main goal of this thesis is to give an overview of the possible ways to handle the previously mentioned problems and to test them within the NEW TIES simulated environment. Chapters 3 and 4 will address the two main objectives:

1. **Target Selection**:

   Design a method, part of a controller architecture, that enables agents to select a target for their actions.

2. **Multi-Objective Controllers**:

   Design an agent-controller-architecture that improves the performance of current reactive-controller-architectures in the NEW TIES environment.

Although both problems seems to be very different, they raise the same questions:

1. Is this a new problem, or are there any methods currently known for solving this problem?
2. Can we introduce new methods?
3. Which of them are most suited to be used within environments with natural selection and learning algorithms that adapt behaviour, like the New-Ties environment?
4. What do we expect to be the effect of the methods
5. Which method performs best within the NEW TIES environment with evolving controllers?
6. How do evolutionary learning mechanisms affect performance?

In this thesis we will address these questions using currently known literature and an evaluation of a set of experiments.

1.5 Structure

To understand the difficulties and opportunities of the proposed NEWTIES controller, we will first explain how agents are controlled now. NEW TIES Agents are using a Decision Q-Tree controller which will be introduced in Chapter 2.

The next two chapters will address the two main objectives of this thesis; Target Selection (Chapter 3) and Multi-Objective Controllers (Chapter 4). Each chapter starts with an introduction and an overview of the related literature. In Section Three, we introduce our own methods. Section Four describes our expectations about the outcomes of the experiments. In Section Four and Five we give the experimental setups and show the results. Each chapter ends with a section describing the conclusions and discussion.

Finally this thesis ends with an overall discussion of the findings and points out future research.

Chapter 2

Decision Q-Tree

From: "Balancing quality and quantity in evolving agent systems" [1]

2.1 Decision Q-Tree

The controller of NEW TIES agents is based upon a decision tree, a so-called decision Q-tree (DQT)*, where each branch in the tree ends with an action and thus can be regarded as a rule deciding on an action. Decision making amounts to traversing this tree in a stochastic manner. Decision Q-Trees have four types of nodes: test nodes, general bias nodes, action bias nodes, and action nodes.

The controller of each agent consist of four parts: the DQT, the Tree Traversal Mechanism, the Target Set Filtering mechanism and the Target Selection Mechanism. The first two are related to the action selection, the second two are related to Target Selection (Chapter 3). The DQT contains the decision-rules (nodes) that the Tree Traversal mechanism uses to determine which action will be selected.

Figure 2.1 shows how a DQT with ordinary tree traversal can be used for choosing the *PICKUP* action. In this example, tree traversal reaches the MOVE action in 50% of all cases, because this is the only defined action for this branch. In the other cases, tree traversal reaches a test node. If the agent sees an object within reach, then tree traversal ends up in the PICKUP leaf. If it does not see any reachable objects, then tree traversal continues to to action bias node. This node has two children, namely TURN_LEFT and TURN_RIGHT. In half of the cases the agent will decide to turn left and in the other cases it will decide to turn right.

2.1.1 Test Nodes

Test nodes are meant to provide a situation description to the agents. A test node evaluates a Boolean query, e.g., "Is there some plant ahead?" or "Is there an agent nearby?", and depending on the answer (Yes or No) the tree is further traversed through either of the two child nodes. In these test nodes concepts are used to describe a certain

*Q hints at the reinforcement learning that implements the individual learning mechanism in NEW TIES

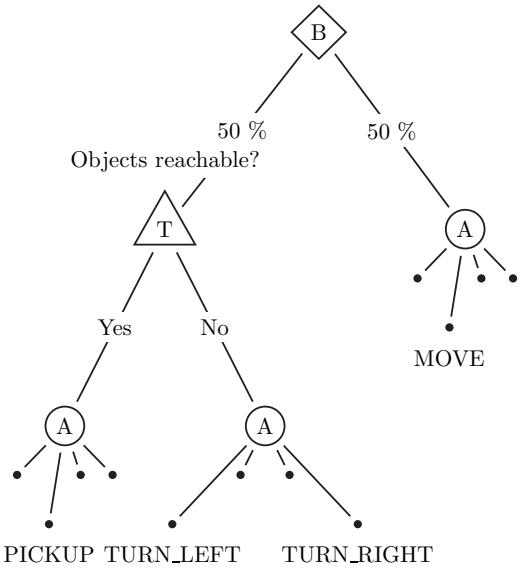


Figure 2.1: Example DQT

Boolean query. For example the test '*VISUAL;PLANT;REACHABLE*' would check if the current agent sees a reachable plant.

Each test has to contain a description of the stimulus-type (Table 2.1) and at least one concept (Table A.1).

Type	Description of content
VISUAL	All objects that are within visual range of the agent
BAG	All objects that are carried by the agent
REPRODUCTION	All agents that proposed a mate to the agent
SELF	The agent itself

Table 2.1: Stimulus types used in the current system

2.1.2 Bias Nodes

Bias nodes facilitate individual choices of the agents driven by their own preferences. A general bias node can be anywhere in the DQT, it may have more than two child nodes, and the choice between the child nodes is probabilistic. The probabilities belonging to each child node $1, \ldots, n$ are calculated by the agent's biases. The probability p_i for choosing child i at a bias node is calculated by the following equation: †

$$p_i = \frac{g_i}{\sum_{j=1}^{n} g_j}, \tag{2.1}$$

where n is the number of edges and $0 \leq g_i \leq 1$.

†Note that we are only using evolution as adaptive mechanism and therefore ignore the learned bias.

2.1.3 Action Bias Nodes

An action bias node is similar to a general bias node except that it is always at the last but one level (above the leaves) and its set of children is the complete set of actions. In all types of nodes the bias belong to the node and if an offspring individual inherits the node, it also inherits the corresponding bias.

2.1.4 Action Nodes

An action node is a leaf node that contains one action. It is carried out provided the environment allows it. The possible actions that can be performed in the current setup of the system are described in Table 2.2.

Action	Description
EAT	Eat a carried plant
MATE	Propose a mate to a visual agent or agree to a proposed mate
MOVE	Move one position forward
NULL	Do nothing
PICKUP	Pickup a visual object within reach
PUTDOWN	Put a carried object down
TURN_LEFT	Turn 45 degrees to the left
TURN_RIGTH	Turn 45 degrees to the right

Table 2.2: Action used in the current system

An example of a valid DQT is exhibited in Figure B.1.

2.2 Evolving Decision Q-Trees

As mentioned in the introduction, NEW TIES is an evolutionary system with asynchronous reproduction and selection. Evolution concerns the inheritable parts of the agents. In the present system only Decision Q-Trees are made inheritable.

2.2.1 Selection operators

Survivor selection

NEW TIES uses a truly environmental selection method, i.e., not based on any task related notion of (centrally calculated) fitness. Agents die if they run out of energy or reach the maximum age M.

Parent selection

In principle, an agent can decide at any time to mate (subject to some constraints). By choosing the action *MATE* it selects itself as a would-be parent. To become a real parent, it needs to find and "convince" another agent of the opposite sex. To do this, it sends a special message, a mate proposal, whose code and interpretation are hard-wired and the

same for all agents. If the other agent accepts this mate proposal the two agents become real parents and produce a child. The environmental constraints on reproduction are that both agents are older than the MateAge threshold value, are of opposite sex and have to be within mating reach.

In order to give a newborn child a viable start, both parents donate one third of their current energy to the child at birth.

2.2.2 Variation operators

Recombination

DQTs are recombined by random subtree exchange as in standard Genetic Programming [4]. In both trees a random crossover point is chosen. The tree of the child is created by taking the tree of the mother and replacing the subtree residing under the crossover point by the subtree under the crossover point in the tree of the father. The genetic biases are simply copied together with the node that the bias belongs to.

Mutation

In NEW TIES mutation complements recombination. Thus, as opposed to usual evolutionary algorithms, mutation cannot be used as a standalone operator to create offspring from a single parent, but always follows recombination. In the present system only one type of mutation operators is used: bias mutation. Bias mutation perturbs the given bias g by a random value drawn from a normal distribution $N(0, 0.5)$, enforcing lower/upper bounds by a simple boundary rule. This mutation operator is applied with a probability of five percent.

Let us note, that in the present version of the system there is no form of bloat control, hence the DQTs in the population can (and will) grow over time.

Chapter 3

Target Selection

3.1 Introduction

The first objective in this thesis is to design methods that enable agents to effectively select an action and a corresponding target. Within NEW TIES, this target selection is crucial for agents to survive. Without any form of target selection picking up food is not possible, which definitive leads to death. The example tree in the previous chapter (2.1) is able to select the *PICKUP* action, however this action fails without any target. The controller has to specify which object has to be picked up, because if there are more than one objects visible, some of them may be more attractive to pickup than others.

In most of the current research performed on emulating artificial societies, Target Selection is not an issue. For example, all experiments performed in Sugar Scape worlds [5] (which is one of the most commonly used environments), agents do not need to distinguish between objects, mostly because they are functional identical in any way, with respect to the agent. Furthermore, actions can be performed on all types of objects and distances are not important. NEW TIES, however, has these constraints which increases the difficulty of the environment. Distance is very important, because some of the visible objects are out of reach and agents first have to navigate toward them in order to successfully pickup an object. Therefore, if the controller decided upon a *PICKUP* action, only reachable objects are attractive. The actions of NEW TIES agents are more like motor actions in robotics, they force an agent to really distinguish between objects and act according to their locations and characteristics.

In the presented experiments, the target selection mechanism has to take care of multiple environmental constraints; It has to be adapted to select that single object that is of the type Plant, non-poisonous, within reach and does not belong to someone else. It has to select this single object from the total set of possible objects that contains all visible objects, like plants and other agents. Every constraint has to be met to prevent the selection of bad targets. Some of these 'mistakes' of the mechanism are more lethal than others. Trying to pickup an agent would simply fail, while picking a poisonous plant can lead to death. It is important that the knowledge about 'good' targets is passed to the offspring of an agent, because there is not much time to unlearn these mistakes themselves.

Furthermore, it is important that the target selection mechanism can cooperate with

the agent controller. The Decision Q-Tree uses the visual information to determine if a pickup action has to be performed and did probably a sequence of tests to check if there are any attractive objects visible. If the Target Selection mechanism does not take this into account, it ignores this attractiveness analysis and agents ends up doing the correctly derived action with the wrong target.

In this chapter we will introduce two new additions to the standard Decision Tree. The first one will be used by the controller to determine the attractiveness of certain objects, while the second addition enables the controller to select the combination of action and target more effectively.

3.2 Literature

Target Selection mechanisms have been studied in the recent and far past by a small amount of researchers. The most common form of target selection is to manually define a function to derive which target is the most attractive to select [6, 7, 8], sometimes combined with a distance function [8]. These mechanisms are quite effective when it comes to solve certain tasks, but are useless within A-Life worlds like NEW TIES. Attractiveness functions are defining and fixing the behaviour of the agent, without any possibilities of adaptive or emergent behaviour. Which is one of the main objectives of the NEW TIES project [9].

To deal with this problem, Tan[10] proposed a method that defines each possible target as a seperate action and applies Q-learning techniques to select such an 'action'. Within NEW TIES the amount of targets is huge and constantly changing. Such a mechanism is therefore not useful.

Spronk e.a. [11] proposed a method that is somewhat more generic than the previous ones. They are using rule sets and dynamic scripting to generate decision rules that contain fuzzy definitions like 'nearest' and 'weakest'. Although their results were very promising, agents are depended of the set of predefined definitions. An algorithm that can be used within NEW TIES has to be able to adapt the targets that are selected, rather than some user defined options.

Bakker [12] introduced a more generic form of this mechanism in his Masters thesis. His mechanism was using a neural network that would select a target for a specified action. His experiments showed that agents managed to select a correct target, but did not adapt this target selection to changing environments. Furthermore the set of possible targets was again very small and not very challenging.

None of the mechanisms found are very suitable for use within NEW TIES, although they can handle the NEW TIES environmental restrictions, they are not flexible enough to handle constantly changing environments and challenges, the big amount of possible targets or do not fit within the NEW TIES objectives.

3.3 Decision Trees with Target Selection

To enable NEW TIES agents to select a target for their actions, we designed two different mechanisms that extent the functionalities of an ordinary decision tree. The first one, Target Set Intersection, is basically an addition to the ordinary DQT and does not change the action selection functionalities (Chapter 2). The second one, Attention Oriented Tree Traversal is introducing a new kind of decision tree traversal that is specifically designed for attention selection, which enables agents to select the combination of targets and actions more effectively.

3.3.1 Decision Trees with Target Set Intersection (TSI)

To be able to select a target, the action selection mechanism is extended to select a target too. Simultaneously with tree traversal, it uses a filtering technique of possible attractive targets. First it creates three sets of targets namely all visible objects, all carried objects and all remembered objects (mate proposals in this case). These three sets are the foundation of the target selection mechanism. Each of the action requiring a target can be mapped to these sets. For example pickup actions can only be performed on visual objects, not on remembered ones. Table 3.1 describes the possible action and target type combinations.

Action	Target type
EAT	Carried
MATE (male)	visual
MATE (female)	Remembered
MOVE	-
NULL	-
PICKUP	Visual
PUTDOWN	Carried
TURN_LEFT	-
TURN_RIGTH	-

Table 3.1: Actions and the corresponding Target types

After this first initial setup, the tree traversal starts at the root node of the decision tree. If this node is a bias node or action bias node, the tree traversal continues to the next node. If the current node is a test node, the target selection mechanism is activated. First it decides if the current test fails or passes. If the test fails, the tree traversal continues with the No-branch, but if the test succeeds, the Target Selection mechanism starts filtering the possible targets. All objects of a certain target type that fail the test will be removed. For example, if the current node contains the test '*BAG;PLANT_TYPE_A*', the set that contains the carried objects will be filtered such that only plants A are left. The same holds for the test '*VISUAL;AGENT;FAR*'. If this test succeeds, the set of visual objects is filtered to contain only agents that are far away, all other visual objects are removed from the set of possible targets. After the filtering, the tree traversal continues with the Yes-branch.

If a second Test node is activated and the test succeeds, the filter is applied on the

current set instead of the whole set. This can result in empty sets, for example if '*VISUAL;PLANT;NEAR*' and '*VISUAL;PLANT;FAR*' and both succeed – the agent observes plants far away and plants nearby – then the set of visual objects has to contain nearby plants that also far away, which is impossible.

Finally, if an action node is reached, the target selection mechanisms determines if the selected action needs a subject. If this is true, the Target Selection Mechanism has to attach an object to this action. In this thesis a Random Target Selection is used, which returns a random object from the corresponding set.

The complete pseudocode can be found in Appendix C.

Example

In this example there is only one agent (A) that has to navigate toward a plant of type A (1 and 2), pick it up, and eat it. It should avoid eating plant type B (3 and 4). The agents is using the DQT displayed in Figure 3.1.

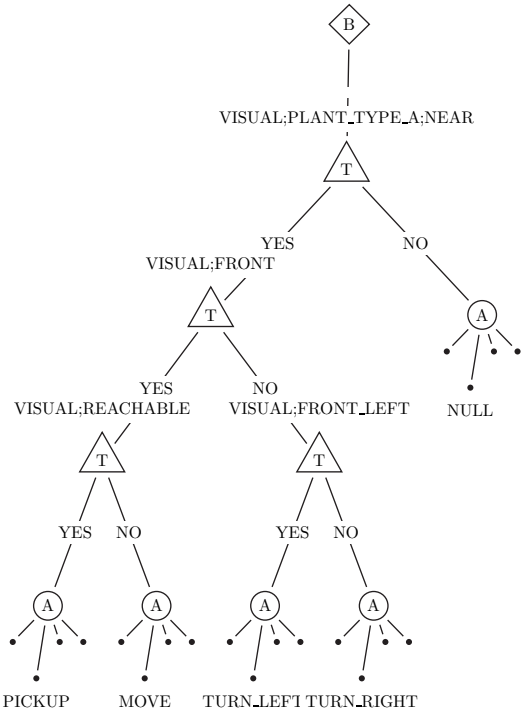


Figure 3.1: Example DTTS

The first node the agent activates is a test node, this node checks if there are any Plant Type A objects visible nearby – which means within a distance of 3. This is true because the agent observes both plant 1, 2 and 3 nearby and plant 1 and 2 are of the correct type. At this moment the set of possible targets contains plant 1 and 2.

The next node is again a testnode. This test also succeeds, because plant 1 and 3 are front. The set of possible targets now contains only plant 1. Plant 3 did not match the first test, plant 2 did not match the second test, and plant 4 did not match any of the tests.

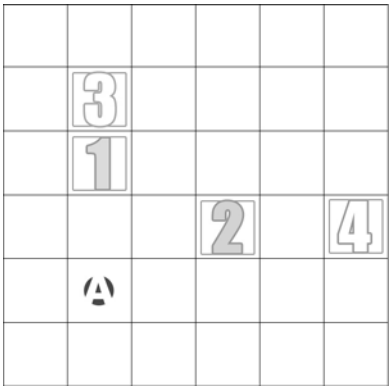


Figure 3.2: Initial situation

The last test checks if there are any plants within reach. This test fails because there is no plant present on the current position of the agent. Now the set of possible targets still holds plant 1, because the set is only filtered when a test succeeds.

Tree traversal reached the end of the tree, and a decision is made to do a *MOVE* action.

In the next timestep, the agent starts again with a complete set of all possible targets. Again it would decide to move forward. In timestep three the agent would decide to do a pickup action with target 1 and the objective is met.

3.3.2 Attention Oriented Tree Traversal (AOTT+TSI)

Decision Trees with Attention Oriented Tree Traversal are very similar to ordinary Decision Trees. The only thing that is different is the evaluation of test nodes. With the ordinary tree traversal, the test is always evaluated on the total set of observations and is not depending on earlier tests. If the tree traversal first test for '*VISUAL;AGENT*' and after that '*VISUAL;RED*', it would succeed in both cases. If the agent observes a red plant and a blue agent. With Attention Oriented Tree Traversal this is not the case, the sequence of test nodes is affecting the evaluations. The success in not only defined by the current test, but concatenates it with all previous successful tests –like the previously mentioned target set filtering mechanism– and evaluates this concatenated test. If the tree traversal first checks for '*VISUAL;AGENT*' and then '*VISUAL;RED*', this second test only succeeds if there are any red agents within the agent's visual field. It is like focusing on a certain type of objects. First the agent wants to know if there are any agents visible. If this is true it would like to know if they are red too. If more tests are added, the agents narrows the set of objects that needs further attention.

The sequence of the test nodes is very important with AOTT, because they define the objects the agent is focusing on. If the agent first test for '*VISUAL;PLANT_TYPE_A*' and then '*VISUAL;PLANT_TYPE_B*' the second one would fail in case the first one succeeded. If the agents now performs a pickup, only plant type A objects are selected. If the sequence of tests is the other way around, again only the first test would succeed, so now only plant type B objects are paid attention to.

The complete pseudocode can be found in Appendix C.

Advantages and Disadvantages

Pinpointing objects

The main advantage of Attention Oriented Tree Traversal is its ability to pinpoint an object and explore its features. This is, for example, very useful for navigation. The following example (Figure 3.3) shows the agent from the previous section in a different setup. The DQT and objectives are still the same.

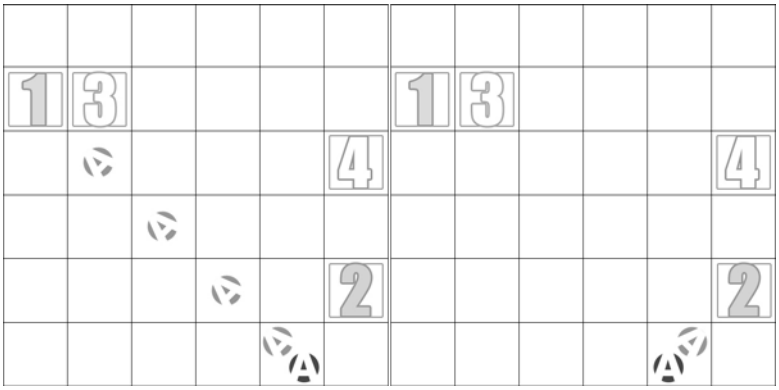


Figure 3.3: Ordinary Tree Traversal against Attention Oriented Tree Traversal

In this example, it is clear that the agent using the Attention Oriented Tree Traversal accomplishes its task much faster. This is because it only paid attention to the nearby plants and therefore ignored plant 1. So if the sequence of tests is correct, the agents using Attention Oriented Tree Traversal would be able to navigate faster and finish the challenge earlier than agents without it.

Cooperation with Evolution

The second advantage of Attention Oriented Tree Traversal is the cooperation with evolution. The Subtree-recombination algorithm can be very destructive to good functioning branches of the tree. This mechanism replaces a certain (sub)branch of the mothers tree with a (sub)branch of the fathers tree. Assume a Decision Tree with Target Set Intersection and a mother with a tree that contains a branch that could perfectly navigate toward a plant and pick it up. If it would add the subtree 'if *VISUAL;AGENT* then *MATE* else *NULL*' to this branch, then this child would not be able to collect anymore while it still carries the correct tree. It would insist on mating because the test '*VISUAL;AGENT*' probably succeeds.

Trees with Attention Oriented Tree Traversal are far less vulnerable to this effect, because the test '*VISUAL;AGENT*' simply fails, because the branch already contained the test '*VISUAL;PLANT*'. This prevents good branches to get invalid due to changes at the leafs. The disadvantage of this is the small influence of fathers genes on the child. If the father brings in a perfect branch, this could be totally ignored by the tree traversal. These branches can only be used if they are inserted on the 'correct' position. Controllers with

ordinary Tree Traversal and Target Set Intersection does not have this problem, these branches are still visited, although target set filtering fails. All actions that need a target would fail, but effective *MOVE* and *TURN* actions are preserved.

3.4 Experimental Setup

We will chose a known problem to represent the learning task, the challenge to distinguish between poisonous and edible plants [13, 14, 15, 16]. In such a scenario there are two types of plans, and both types of plants can be picked up and eaten by the agents. Eating edible plants gives the agent energy, while poisonous plants will drain energy. In this study we scaled the system so that a poisonous plant drains one and a half times the energy than an edible plant yields. The objective is to eat as much edible food as possible. This objective implies learning to distinguish bewteen the two types of food, because otherwise the population dies prematurely due to the poisonous food. The choice for this challenge is motivated by two properties that help to monitor our simulations. First, the effectiveness of the behavior of the agents influences the environment in a direct and visible way: the number of edible/poisonous plants can be easily tracked. Second, it is easy to monitor the 'learning' behavior of the agents by keeping score of the number of each type of plant they eat.

In the experiments reported here, the value for MateAge is equal to 10% of the maximum lifetime M. The value for MaxEnergy is set at the amount approximately needed to survive 72 random eat actions.

We use a world that is a grid of 400x400 cells and a maximum of 160.000 plants is enforced because of computational resources. Likewise, we limit the total number of agents by preventing the initiation of new MATE actions after the population size reaches 2000.

At the start of each experiment the world is initialized with 500 agents and approximately 16.000 plants of each type, all randomly distributed over the grid. Initially, all agents have the same controller as shown in Figure D.1. Each simulation runs for 20000 time steps, which was found a sufficient number of time steps to learn the task and finish the objective.

3.5 Performance Measures

To compare the performance of both methods we are using the following measures:

1. The average amount of timesteps until $G = 0.99$
   G is defined for a certain timestep as: $\frac{\#\ eaten\ plant\ A}{\#\ eaten\ plant\ A + \#\ eaten\ plant\ B}$
2. How many experiments reach a P-value of 0.95.
   P is defined for a certain timestep as: $\frac{\#\ plant\ B}{\#\ plant\ A + \#\ plant\ B}$
3. The average amount of timesteps until $P = 0.95$

4. The maximum value of E
   E is defined for a certain timestep as: $\frac{\#\ eaten\ plant\ A}{\#\ population\ size}$

5. The maximum value of E_2
   E_2 is defined for a certain timestep as: $\frac{E}{\#\ plant\ A}$

The G measure gives an indication of the amount of time, agents need to learn eating only edible plants. The P measures indicate how if and fast the agents manage to overgraze the world. Measures E and E_2 indicates how effective a single agent is eating. The first 2500 timesteps are ignored in these last two measures because of initialization effects, furthermore all points are removed where the population is smaller than 50. These small populations are causing a great disturbance of the measures because of the high fluctuations in the amount of eat actions and the corresponding measure.

3.6 Expectations

Agents with and without Attention Oriented Tree Traversal will have the same performance when it comes to learning the task, because they both can learn it perfectly. Agents would eventually learn to eat only edible food until the food is exhausted. The big difference can be expected in the time that agents need before they learn to eat only the edible food, and the time that it takes for those agents to eat all edible food available.

With Attention Oriented Tree Traversal, evolution is slowed down due to the small effect of father genes and the importance of the sequence of tests. It is therefore likely to be more sensitive than simulations with ordinary tree traversal, because learning the task is more difficult. Furthermore the effectiveness of navigation affects evolution. Agents with AOTT will navigate and eat more effective which increases the survival pressure. This can lead to faster learning, but if it is to hard it can also lead to the death of the population before the task is learned.

The race to eat all edible food in the world (overgrazing) is probably won by agents with AOTT. Agents with AOTT will be able to adapt a tree that can effectively navigate and pickup objects. This gives them a huge benefit over agents without it.

3.7 Results

The results are shown in Table 3.2 which contain the success criteria. Furthermore, Figure 3.5 to Figure 3.7 show the four different measures. Each graph contains two lines, each line representing one of the target selection mechanisms. All the results are an average of 40 runs.

A remarkable result is shown in Figure 3.4. This figure shows the learning behaviour of the agents. Although we expected that agents with Attention Oriented Tree Traversal is slower on learning, because it is sequence sensitive, we observe no noticeable difference.

Criteria	DTTS	AODT
The average amount of timesteps until $G = 0.99$	5054	4726
The number of experiments that reach $P = 0.95$	25	28
The average amount of timesteps until $P = 0.95$	13671	10939
The maximum value of E	0.00742	0.00930
The maximum value of E_2 (x 10.000)	0.00270	0.00406

Table 3.2: Results

Kolmogorov-Smirnov Tests also indicate that there is no significant difference* between the speed of learning.

All other measures indicate a better performance with Attention Oriented Tree Traversal on this task. In 70% of the experiments with Attention Oriented Tree Traversal, the agents manage tot eat almost all edible food, against 62% without it. Not only the success rate is higher, but the performance is better too. It took AOTT-agents on average 10939 timesteps until P was above 0.95, while agnets without AOTT needed 13671 timesteps.

The high success rate and fast performance is caused by the effective movement of the AOTT agents. As shown in section 3.3.2, AOTT agents should be able to navigate more efficient. Figure 3.6 and Figure 3.7 show that this effect indeed occurs; AOTT agents perform significantly more efficient on collecting plants than agents with ordinary tree traversal.

In this case, using the averages of the two mechanisms is not enough to judge which one in best. Although AOTT agents seems to perform much better, it is also the most sensitive one. To illustrate this, Figure 3.8 is showing the P measures of the best and the worst five runs of both setups. Interesting is that the five best runs with AOTT perform equal to the best three runs without it, but that the worst experiment with AOTT performed dramatically less than all others. In this experiment the agents managed to learn the difference between both plants quite fast but did not evolve an efficient navigation subtree. This is probably caused by the fact that agents could survive with a small amount of food as long as it is edible.

*Using a α of 0.05

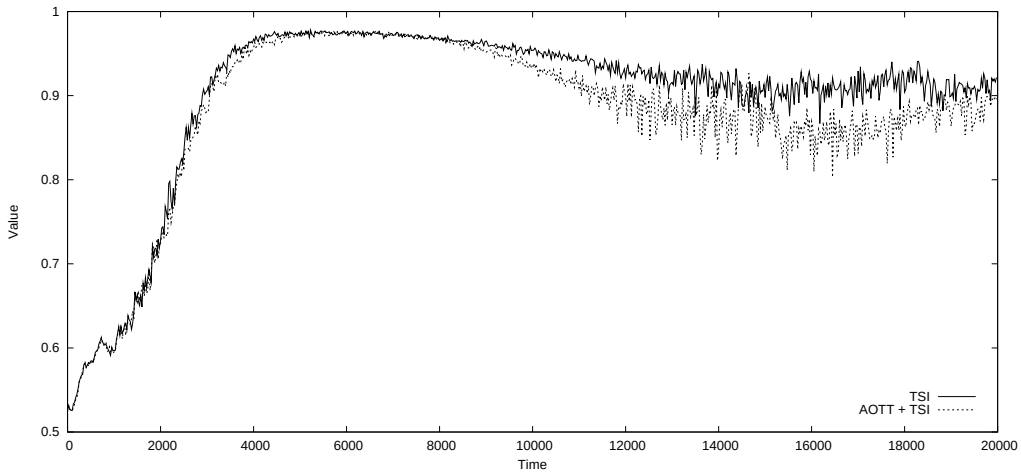


Figure 3.4: G-measure

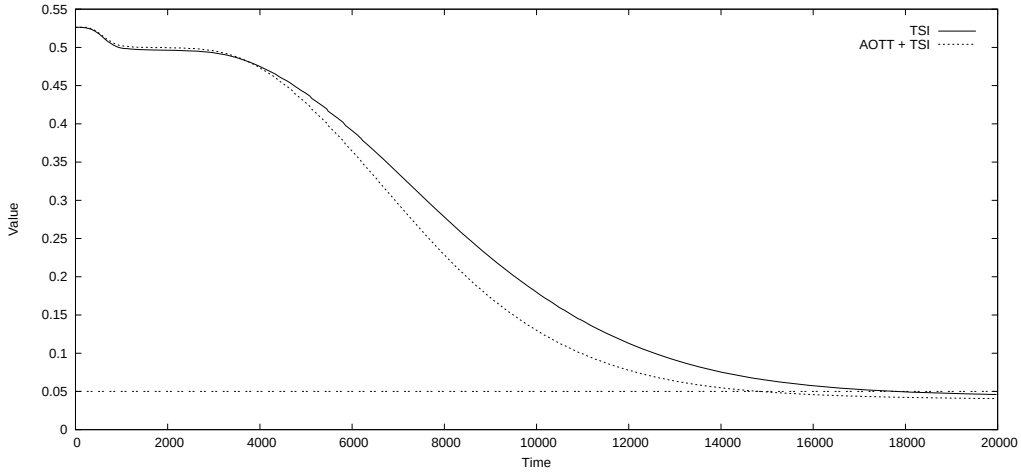


Figure 3.5: P-measure

3.8 Conclusions

We can conclude that agents with Attention Oriented Tree Traversal can adapt more efficient navigation skills than agents with a ordinary Tree Traversal. Experiments show that none of the two proposed mechanisms performs better on learning a certain task and therefore efficiency is the distinguishing factor.

Although it is more sensitive, combining Target Set Intersection and Attention Oriented Tree Traversal would probably lead to the highest performance in NEWTIES experiments. Especially when it is important to collect efficient, for example when plants give less energy than in the presented experiments, Attention Oriented Tree Traversal can be the key to survive. Challenges in which navigation and objects distinction are not the main issues and therefore do not need very effective collecting behaviour, can benefit from the stable performance of agents with ordinary tree traversal.

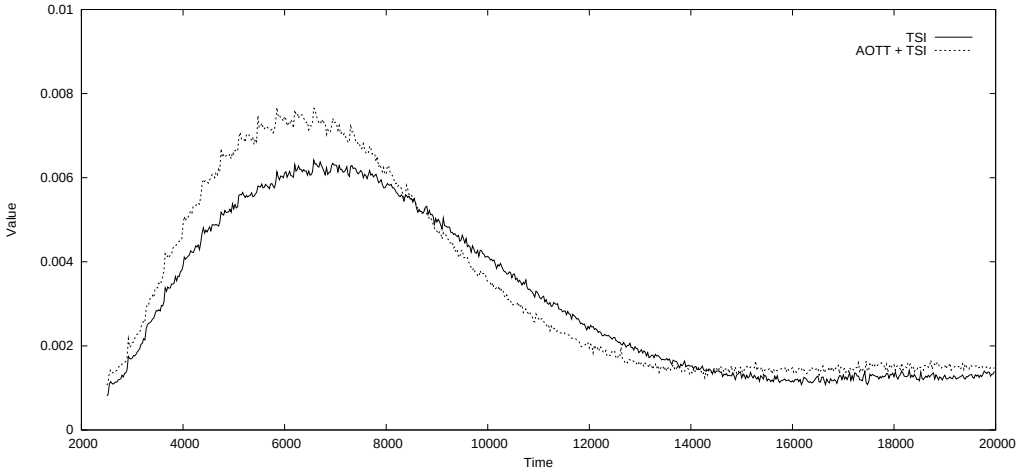


Figure 3.6: E-measure

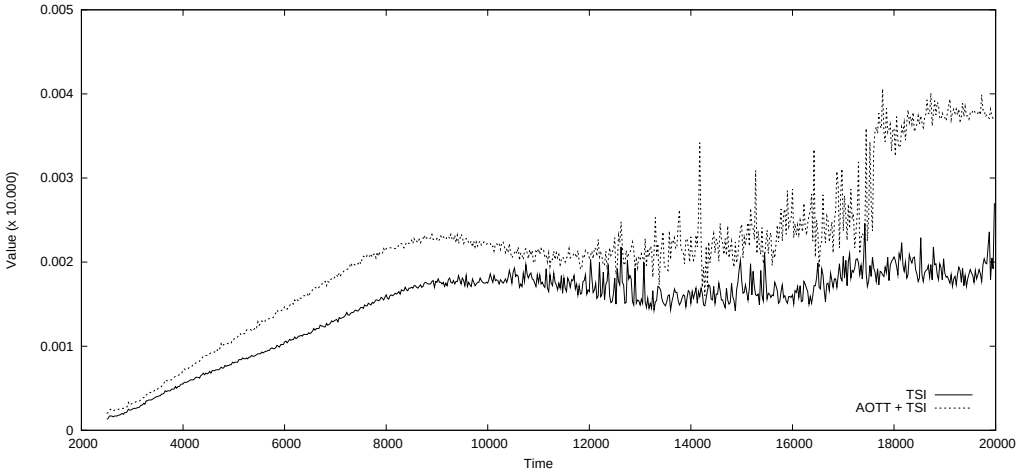


Figure 3.7: E_2-measure

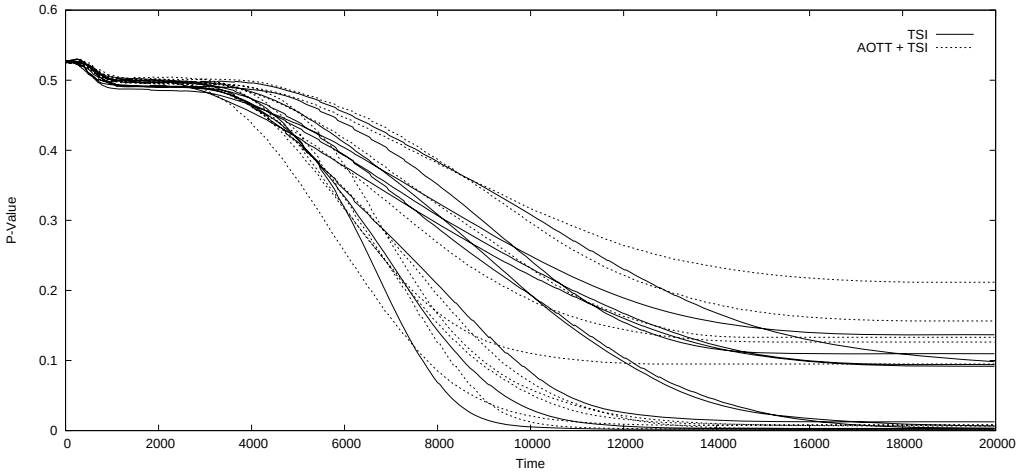


Figure 3.8: Best and Worst runs of both mechanisms

Chapter 4

Multi-Objective Controllers

4.1 Introduction

Decision Trees are very suited when it comes to reactive behaviour, but many of the more interesting challenges cannot be solved by simple reactive behaviour. In Chapter 1.3 we described an example of a multi-objective challenge. The agent not only has to stay alive, but it has to run as far as possible too. With reactive behaviour, the agent continuously repeated the inefficient behaviour of eating one bite of food and running a few meters. A far more efficient behavior is to eat until it is fully satisfied and then run until it is hungry again. This behaviour cannot be performed by a reactive controller, due to the lack of state-persistency. In this example, using reactive behavior is only inefficient, but in some cases handling multiple contradicting objectives is needed for survival of its species. Such a challenge is described in Section 4.4.

The second objective of this thesis is therefore: to provide the agents with a capacity that can deal with (contradicting) multi-objective challenges.

Much work has already been dedicated to this topic, but there is not much research carried out on the possibilities of applying evolution on controllers that support non-reactive behaviour. This ablity to learn and adapt its behaviour without any interference of the developer is the main objective of the NEW TIES project and therefore the main restriction to the controller. Furthermore, agents have to be able to use and adapt this behaviour without being restricted to the thoughts of the scientist, otherwise emergent behaviour is less likely to emerge.

There are more problems the controller has to deal with. Unlike agents in many other environments, the only selection on 'good' and 'bad' agents in NEW TIES, is by means of natural selection. There is no explicit fitness function or other way to define beforehand which agent performs well and which is bad. This results into a much smaller selection pressure than used in most other experiments. In NEW TIES challenges, it is not known beforehand which behaviour is good. The only task for each agent is to pass down its genes to its offspring. Only the pool of genes at the end of the simulation indicates the fitness of previous generations. The environment and therefore the problems agents face, changes continuously during the simulation. Agents modify their own environment, including themselves, creating new opportunities and tasks. The controller has to be able to deal with these dynamics and uncertainties.

In this Chapter we introduce a method that enables adaptation and tuning of the agent behaviour by Genetic Programming. This method has to be embedded in the current controller such that they can give agents the benefit to solve multi-objective challenges. The challenge as described in Section 4.4 can not be solved by the ordinary Decision Tree and requires some fundamental changes to the controller.

4.2 Literature

Much research has been performed on controllers that can handle multi-objective environments. The common factor of this research is the general idea behind the solution: Each of the researchers propose their own variation of the same idea. Agents should be able to perform a certain task until the controllers tells them to do another task. We will therefore, use this idea of persistence as a base for our mechanism too. The persistence of the execution of a certain task is implemented in many different ways. Some of the researchers[17, 18] use controllers that are similar to our Decision Tree. Others use Neural Networks controllers [2, 19, 20, 21, 22], Learning Classifiers[23] or other types of controllers [24]. In the following two sections, an overview is given of the possible implementations of a non-reactive agent. First we will describe the possible DQT-like solutions and after that we will discuss some of the other solutions that, due to the type of controller, cannot be directly incorporated into our agents but contain ideas that can enhance them.

4.2.1 Decision Trees

The research of Grefenstette[17] and Bryson[18, 25] is the most related to the multi-objective problem. However, both are applied to a specific field and their ideas are therefore too specific to work on our agent. A successful NEWTIES agent should incorporate the ideas of both researchers.

Grefenstette is describing experiments in which the controllers are adapted by learning and evolution in almost the same way as NEWTIES does. In the context of NEW TIES, his *"plans"* can be regarded as sequences of test and action nodes describing a certain behaviour. His experiments show that it can be favorable to crossover on the probability to activate a plan, instead of evolving the plan itself. There are several reasons why Greffenstette's ideas cannot straightforwardly be applied to the NEW TIES decision trees. First of all, these experiments were performed using a fitness function to define which agents are 'good' and 'bad' instead of natural selection that does not possess an explicit fitness function. Furthermore, agents did not chose to reproduce themselves. Greffenstette used an oracle to define which agents creates offspring. These are fundamental differences with the NEW TIES environment and the results can therefore be very different. Finally, the agent structure of Greffenstette is purely reactive and therefore cannot deal with multi-objective environments, as described in the previous section.

Bryson introduced a very effective agent architecture that enables agents to survive very challenging environments with multiple objectives. A drawback of this architecture is

that it is not directly suitable for evolution and learning. Although her architecture could be easily adapted to fit into the current NEW TIES agent structure, it is not known how sensitive it is with respect to manipulations by learning algorithms and evolution.

4.2.2 Other controllers

There is only a very small number of researchers that investigated the complete spectrum of the problem our agents face. These researchers mostly define two layers of control. Behaviours are used to determine what the response would be if an agent has a certain behaviour. The top layer keeps track of these responses and determines what behaviour should be activated. This could be done with activation functions [2, 23, 20], action voting[19] or emotions[21]. Although the main idea of separating the tasks in individual behaviours can be benefitial to our agents, most of this research cannot be applied to our problem.

The problem with these methods is the presence of a fitness function that defines which agents survive and reproduce. The NEWTIES environment does not have such an explicit fitness function, the only available section mechanism is natural selection. Although 'good' agents probably survive and 'bad' agents probably die, it is not guaranteed that the pressure of natural selection is strong enough to create the same results. Furthermore, the absence of lifetime learning in in the current setup would reduce the performance of these methods dramatically. The research carried out by Dorigo [23] is very similar to our research, but he rewards certain behaviors if they finished a certain task successfully. By using these specialized rewards for executing a certain task, behaviours are tuned for a certain predefined task. Furthermore, he uses an explicit fitness functions to define which agents survive and reproduce. Without this way of discriminating good agents from bad agents and specializing behaviours to do a certain task, his agents would not show the excellent behaviour as described in his research. This lack of discrimination possibilities makes our problem new and challenging.

4.3 Behaviour Oriented Decision Trees (BODT)

The previous chapter already showed that Decision Trees, used with Genetic Programming, can be effective in Artificial Life simulations. It would therefore be logical to keep the existing NEWTIES controller architecture and extend it to enable agents to deal with multi-objective environments. This altered architecture is very similar to the "Behaviour Oriented Design" Bryson introduced. Furthermore, we introduce in our architecture some features from the research of Sengers[26]. Sengers introduced a structure of different types of behaviours * that enables switching and returning. We combined these two structures with the current DQT architecture and introduce a new variation to the Decision Tree; the Behavior Oriented Decision Tree. If the task agents face can be decomposed into several layers of tasks, Behaviour Oriented Decision Trees enable the designer to decompose the controller too.

*In this case 'behaviour' does not refer to the externally observable behaviour, but describes a sequence of decisions and actions that is made during the execution of an certain activity.

4.3.1 BehaviourSwitch and BehaviourReturn nodes

Controllers with Behavior Oriented Decision Trees consist of a set of ordinary DQTs that can be seen as Brysons Competences. They represent a certain knowledge or skill. Test nodes represent the different tests that are applied to determine which action should be chosen, like conditions do in BOD. To enable competences to activate another competence, we introduce two new nodes: BehaviourSwitch and BehaviourReturn. At the moment a BehaviourSwitch node is activated it moves the tree traversal to the root of another DQT. BehaviourReturn nodes are used to switch back from the currently active DQT to the previous one at the moment the goal of the task is reached. These functionalities are illustrated in Figure 4.1. In this example BODT solves the problem described in section 1.3.

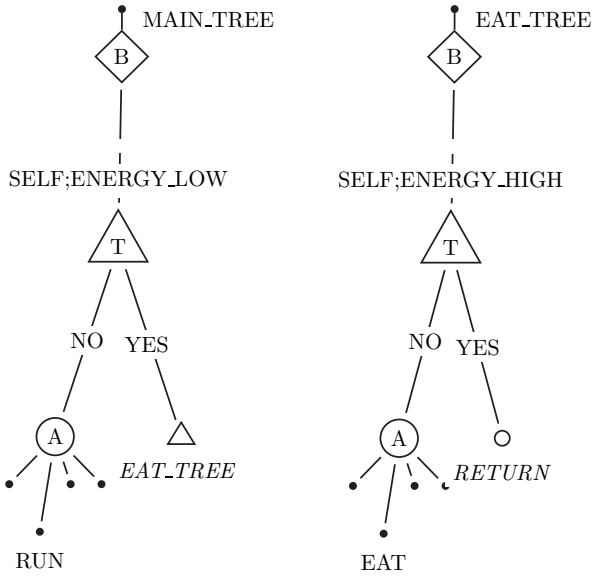


Figure 4.1: Example of BehaviourSwitch and BehaviourReturn

The tree traversal starts in the *MAIN_TREE* and continues to return *RUN* actions as long as the energy keeps above the *ENERGY_LOW* threshold. When the energy drops below this threshold the tree traversal continues to the BehaviourSwitch node (triangle). This node replaces the *MAIN_TREE* with the *EAT_TREE* and continues traversing. From now on the tree traversal starts with the *EAT_TREE* and therefore returns *EAT* actions until the *RETURN* node is activated. This node reverts to the root of the *MAIN_TREE* and activates it.

This BODT causes the desired behaviour: The agent keeps running until it is hungry, then it would eat until it is full and starts running again.

4.3.2 BehaviourStart node

In "The Behavior-Oriented Design of Modular Agent Intelligence" [25], Bryson points out the need for another feature in BODTs. BODTs have to contain a mechanism that deals with alarms, request or opportunities that make pursuing a different plan more relevant. She called this mechanism a *Drive Collection*. It contains all tests that are always checked before the ordinary tree traversal continues. This collection can contain, for example, rules that trigger when there is a threat nearby and activate an avoidance behaviour.

To incorporate this feature in the BODT we introduced a third type of node: the BehaviourStart node. In the previous example a case was illustrated where the complete *MAIN_TREE* is replaced by the *EAT_TREE* and hereby losing any test that indicated threats. Figure 4.2 illustrates this with the running agent example, with an additional rule that tells the agent to avoid dogs.

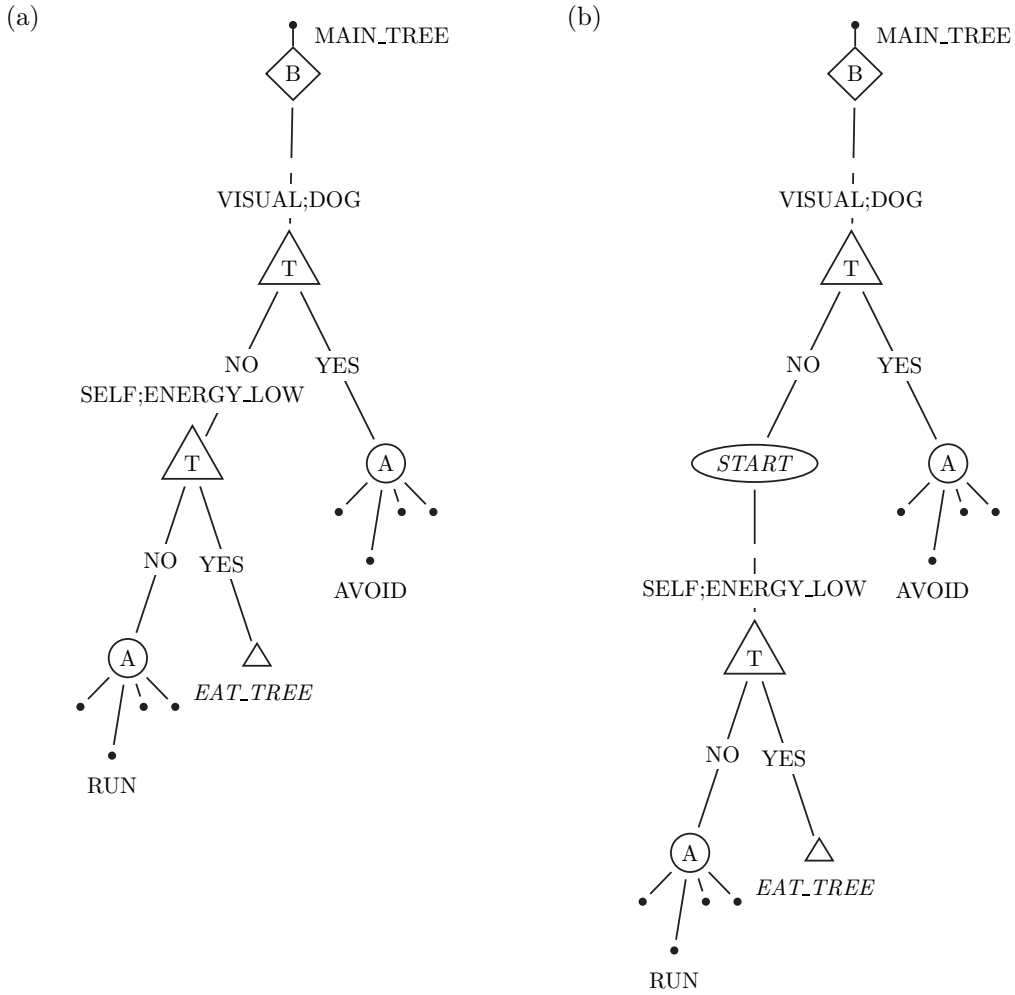


Figure 4.2: DQT with and without BehaviourStart node

The first BODT (Figure 4.2a) causes the agent to avoid the dog as long as the agent is running, but it will not avoid the dog when the agent is eating. The *"VISUAL;DOG"* test

is replaced after the *EAT_TREE* is activated and is therefore not tested. To deal with this problem the BehaviourStart node is used. When the tree traversal reaches a BehaviourSwitch node, instead of the complete tree, only the part between the BehaviourSwitch node and the BehaviourStart node is replaced. This mechanism can therefore protect certain tests for being replaced when switching trees by inserting a BehaviourStartNode. To solve the problem of the extended running agent problem a BehaviourStart node has to be inserted between the "VISUAL;DOG" and "SELF;ENERGY_LOW" tests. This ensures that "VISUAL;DOG" is always tested (Figure 4.2b).

These three additional nodes give the controller the opportunity to create and use behaviour switching to deal with multiple contradicting objectives in an environment. Furthermore, learning mechanisms and evolution can still adapt trees in the same way as ordinary DQTs. Only recombination on the set of trees requires some design choices. In this thesis, we have chosen to apply recombination only on the main tree as Grefenstette [17] suggested, other possibilities are discussed in Chapter 6.

4.4 Experimental Setup

To test the performance of agents that are capable of behaviour switching with agents that are purely reactive, we altered the running agent example to fit into the NEWTIES environment. Agents still have to collect food in order to stay alive, but now agents face another task. They are only able to mate if they carry at least three tokens in their bag. These tokens are distributed all over the world and they start decaying while being carried. Agents will only have limited time to collect them and find a mate which is also carrying enough tokens.

In the experiments reported here, there is no minimal age for mating and newborn agents can live forever. The initial population of 500 agents can live for 5 to 25 years and is supplied with enough energy to create at least three generations of offspring in order to generate enough diversity. The number of agents is restricted to 2000, due to computational resources.

The value for MaxEnergy is set at the maximum amount of energy needed to collect three tokens, reproduce and collect food again. Agents can check their own energy level by means of three different states: LOW, MEDIUM, HIGH. The threshold for LOW is set to the minimum amount of energy needed to find food in most cases. The HIGH threshold is set to the average amount of energy needed to collect three tokens and reproduce without getting below the LOW threshold.

We use a world that is a grid of 160x160 cells and contains 5000 plants. Each plant regenerates within 10 timesteps after being picked and gives enough energy to, on average, perform 10 actions. By using regenerating plants instead of the ordinary NEWTIES plant growth and reproduction we can prevent overgrazing and undergrazing effects. The world contains 600 tokens. Each token regenerates within 20 timesteps after being picked, while picked tokens will decay within 100 timesteps and disappear from the bag. These settings create an environment that, in an optimal situation, takes agents only 20 timesteps to collect three tokens.

Initially, all agents have the same reactive controller as shown in Figure D.2. This initial tree is reactive due to the fact that the accessible BehaviourSwitch node, that activates the *MATE_TREE*, is directly preceded by a BehaviourStart node. The learning task is to create a tree in which the BehaviourSwitch is not preceded by a BehaviourStart node, enabling the agent to execute the *MATE_TREE* until the agent is low on energy. The *EAT_TREE* and *MATE_TREE* will be copied to the child, while the *MAIN_TREE* will be a recombination of the trees of both parents. Furthermore, this tree is adapted by mutation. These two operators enable the creation of agents that are able to use the full functionalities of BODT.

To be able to judge about the benefits of BODT we carry out another set of experiments. The agents from these experiments can not use the main benefits of BODT: the replacement of trees is turned off. This results in an ordinary DQT with an additional feature that protects certain parts from being recombined or mutated. When the tree traversal activates a BehaviourSwitch node, the traversal continues with the subtree without replacing the main tree.

Each simulation runs for 100.000 time steps, which was found a sufficient number of time steps for the newborn agents to reproduce for several generations.

4.5 Performance Measures

To compare the performance of agents that can learn to use BODTs with agents that cannot, we are using the following measures:

1. The population-size at timestep 100.000
2. The maximum generation reached
   The generation of a child is defined as the maximum of the generations of its parents plus one

The first measure indicates how well the population manages to overcome the mating restrictions, while the second measure indicates how well the solution to these restrictions is passed to the offspring.

4.6 Expectations

We expect that agents with the full functionalities of BODT will perform much better than agents without them. It would take agents at least 20 timesteps to collect three tokens, so reactive agents start gathering food again before they can reproduce. This probably results into the decay of the tokens before the agent manages to get three of them. BODT agents do have enough time to collect them, because they can learn to start eating again at the LOW threshold instead of the MEDIUM one.

The only thing that can happen is that none of the newborn agents has an effective BODT. If an agent cannot gather three tokens, they will not reproduce. The mate-selection pressure is therefore very high. This can cause populations to die, if there are not enough good agents. On the other hand, if enough agents have an effective tree, this tree is probably propagated to the offspring, because both parents have one. If, at some point in time, the population begins to increase, then the population would probably start increasing faster and faster, until there are not enough tokens or food to support such a big population, or the maximum population of 2000 is reached.

4.7 Results

The results are shown in Figure 4.3 and Figure 4.4. As expected, the agents that can use the full functionalities of BODT are outperforming the ones that cannot use it. Without BODT, the population size did hardly increase. These agents did not learn to switch correctly between the two tasks, which was expected. The agents that were able to learn to use the full functionalities, did.

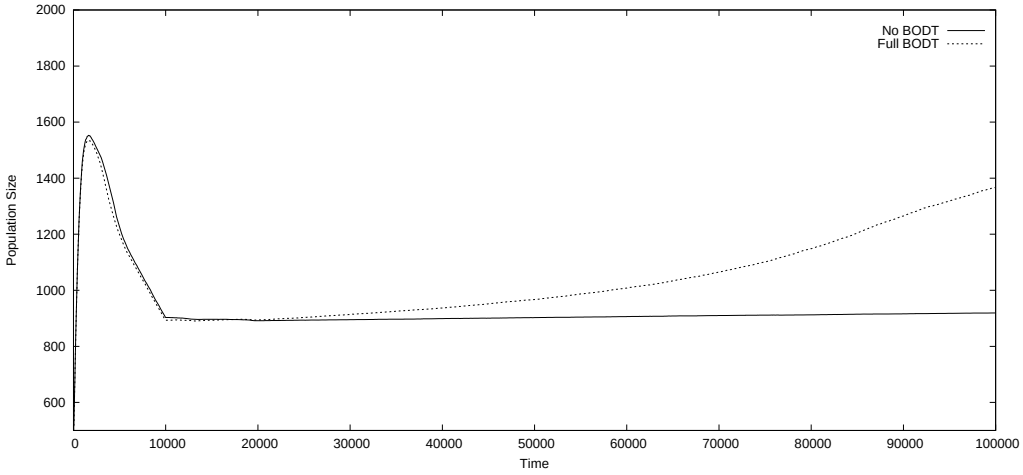


Figure 4.3: Population size

In the first 10.000 timesteps, some of the offspring possesed a tree that switched efficiently between the two behaviours. These agents managed to find each other and reproduced. This offspring also received an efficient tree and started to reproduce, creating new generations of agents. Figure 4.4 illustrates that agents with efficient BODT trees passed down their genes for many generations, while agents without BODT stopped reproducing after the initial phase. Figure 4.5 illustrates how the new generations of agents using BODT kept reproducing over time. Evolution kept preserving the essential part until the end of the experiment.

Although some of the experiments ended with only a small population increase (Figure 4.6), all of them terminated with an increased population. We found that the number of timesteps needed for a population growth of 100 individuals is an exponential function (Equation 4.7 and Figure 4.7) based on the current population size. The true initialization point is therefore not timestep 0, but timestep 10.000. The different population

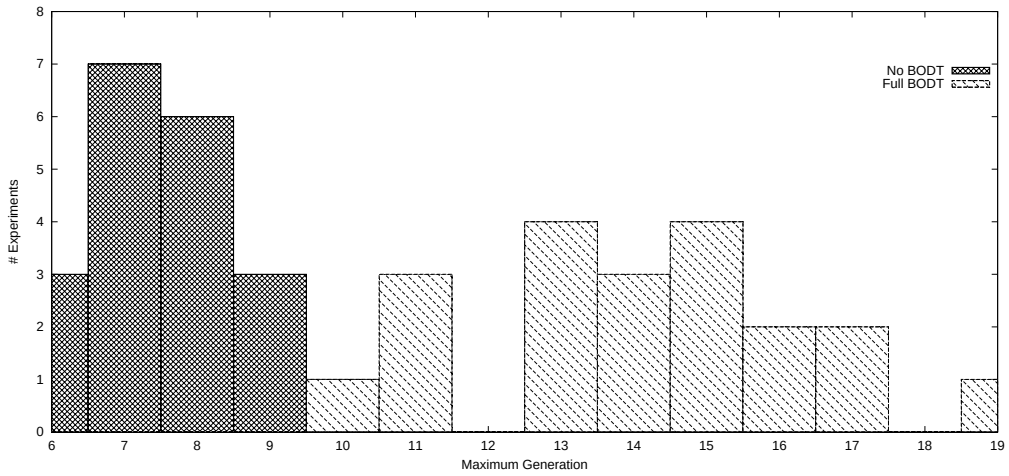


Figure 4.4: Histogram of Maximum Generation reached

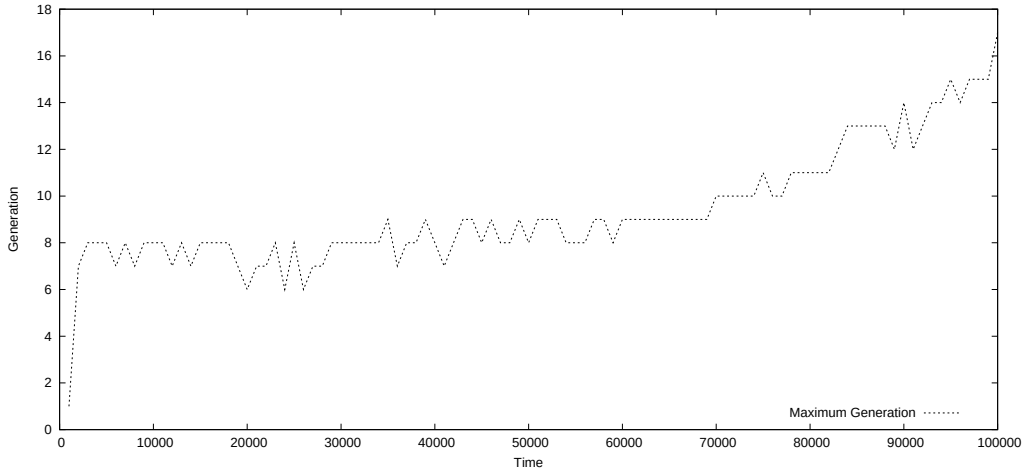


Figure 4.5: The maximum generation over time

sizes at that point determine the growth of the population during the experiment. From that point on, the population sizes start to increase exponentially with the same ratio. The function 4.7 fits all experiments with no significant differences at low population sizes (except for one experiment), which means that the growth of a population can be predicted and is hardly influenced by other factors than the population size itself. Differences are found in large populations, which are caused by the limited number of tokens and plants in the current environment.

$$pop(x + 100) = 30.000 \cdot e^{-0.0034 \cdot pop(x)} \tag{4.1}$$

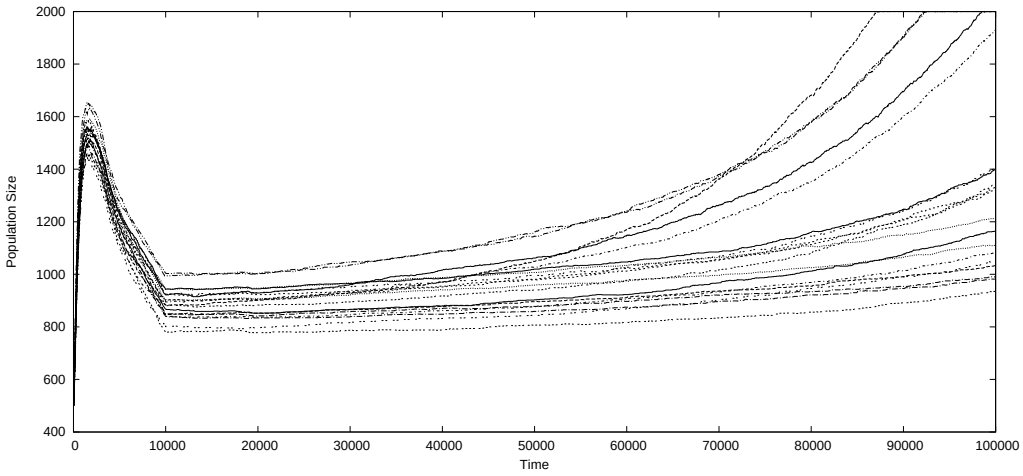


Figure 4.6: The Population Sizes of all full BODT experiments

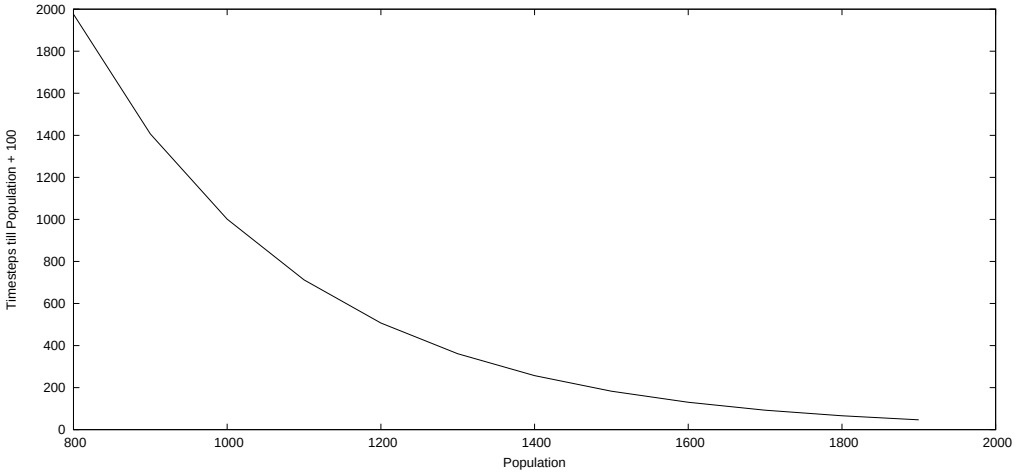


Figure 4.7: Timesteps needed to increase the population, based on Equation 4.7

4.8 Conclusions

We can conclude that Behaviour Oriented Decision Trees can be used to handle problems related to multi-objective environments. Furthermore, experiments show that BODTs can cooperate with evolutionary algorithms. Agents are not only able to use Behaviour Oriented Decision Trees effectively, but can also evolve trees that enable such behaviour. Evolution introduces and preserves the essential parts of the BODT. This is mainly caused by the fact that in these experiments, a high pressure on mating is executed. Only agents that can use BODTs effectively can reproduce. The size of the 'good' population is therefore very important. If the population contains only a small amount of 'good' agents, then the the population increases only a little. It takes relatively more time for these populations to increase with a certain amount of agents than populations with a high amount of 'good' agents. But if a critical mass starts to reproduce, then the population starts increasing exponentially.

Chapter 5

Overall Conclusions

From the Target Selection experiments, we can conclude that agents with Attention Oriented Tree Traversal can adapt more efficient navigation skills than agents with a ordinary Tree Traversal. Experiments show that none of the two proposed mechanisms performs better on learning a certain task and therefore efficiency is the distinguishing factor.

Although it is more sensitive, combining Target Set Intersection and Attention Oriented Tree Traversal would probably lead to the highest performance in NEWTIES experiments. Especially when it is important to collect efficient, for example when plants give less energy than in the presented experiments, Attention Oriented Tree Traversal can be the key to survive. Challenges in which navigation and objects distinction are not the main issues and therefore do not need very effective collecting behaviour, can benefit from the stable performance of agents with ordinary tree traversal.

From the Multi-Objective Controller experiments, we can conclude that Behaviour Oriented Decision Trees can be used to handle problems related to these kind of environments. Furthermore, experiments show that BODTs can cooperate with evolutionary algorithms. Agents are not only able to use Behaviour Oriented Decision Trees effectively, but can also evolve trees that enable such behaviour. Evolution introduces and preserves the essential parts of the BODT.

Chapter 6

Discussion

Both Attention Oriented Tree Traversal and Behaviour Oriented Decision Trees proved their value in the previous chapters. The populations managed to adapt themselves to the challenging environments. Parents passed their 'knowledge' to their offspring, giving them a better chance to survive. What happened was fascinating, this offspring did not only dealt with the problem, they also created a new problem. From that point on, agents did not only needed to distinguish between plants they also needed to navigate effectively too. Effectively eating agents were overgrazing the world, causing less skilled agents to die. The offspring, with a better skill, increased the selection pressure on the whole population, creating a form of co-evolution.

Just like humans, these agents changed their environments, introducing new challenges to survive. Creating such forms of emergent behaviour is the final goal of the NEW TIES project. Agents should have space to develop themselves in a way, not bounded to any design choice or other constraints. Therefore it is desirable that none of the behaviours is hardwired or protected against learning and evolutionary algorithms. Dropping these restrictions brings new problems, huge decision trees are easily destroyed and evolutionary learning is slowed down. Introducing separate trees decreases the size of an individual tree, but creates bounds to the tasks a certain behaviour can solve. Tree traversal techniques, learning mechanisms and evolutionary operators have to work together to prevent good behaviours to be broken without adding constraints to the behaviour of the agents themselves. Only such kind of agents can emerge intelligent behaviour.

Bibliography

[1] A.E. Eiben, A.R. Griffioen, J.A. Bekker, and E.W. Haasdijk. Balancing quality and quantity in evolving agent systems. Submitted to: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'07), 2007. [cited at p. iv, 5]

[2] B.M. Blumberg. *Old tricks, new dogs: ethology and interactive creatures*. PhD thesis, Massachusetts Institute of Technology (MIT), 1997. Supervisor-P. Maes. [cited at p. 1, 21, 22]

[3] A.E. Eiben, A.R. Griffioen, and E.W. Haasdijk. A generic system for studying population-based adaptive systems. To Appear in: Proceedings of the European Conference on Complex Systems 2007 (ECCS'07), 2007. [cited at p. 2]

[4] W. Banzhaf, P. Nordin, R.E. Keller, and F.D. Francone. *Genetic Programming: An Introduction*. Morgan Kaufmann Publishers, Inc., 1998. [cited at p. 8]

[5] J.M. Epstein and L.R. Axtell. *Growing Artificial Societies: Social Science from the Bottom Up*. The MIT Press, November 1996. [cited at p. 9]

[6] D.C. Han and K.S. Barber. Desire-space analysis and action selection for multiple dynamic goals. In *Proceedings of 5th International Workshop on Computational Logic in Multi-Agent Systems*. [cited at p. 10]

[7] Sanguk Noh and Piotr J. Gmytrasiewicz. Multiagent coordination in antiair defense: A case study. In *Modelling Autonomous Agents in a Multi-Agent World*, pages 4–16, 1997. [cited at p. 10]

[8] P. Cristea, A. Arsene, and B. Nitulescu. Evolutionary intelligent agents. In *Proceedings of the Congress on Evolutionary Computation 2000*. IEEE Computer Society Press, 2000. [cited at p. 10]

[9] N. Gilbert, M. den Besten, A. Bontovics, B.G.W. Craenen, F. Divina, E.A. Eiben, R. Griffioen, G. Hévízi, A. L orincz, B. Paechter, S. Schuster, M.C. Schut, C. Tzolov, P. Vogt, and L. Yang. Emerging artificial societies through learning. *Journal of Artificial Societies and Social Simulation*, 9(2):9, 2006. [cited at p. 10]

[10] M. Tan. Learning to coordinate for target selection. Masters Thesis, Supervisor: T. Öztürk, Middle East Technical University, 2003. [cited at p. 10]

[11] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma. Online adaptation of computer game opponent AI. In *Proceedings of the 15th Belgium-Netherlands Conference on Artificial Intelligence*, 2003. [cited at p. 10]

[12] J. Bakker. Online adaptive learning in a role playing game. Masters Thesis, Supervisor: M. Wiering, University of Utrecht, 2004. [cited at p. 10]

[13] S. Nolfi and D. Parisi. Learning to adapt to changing environments in evolving neural networks, 1995. Nolfi, S., and Parisi, D. (1995). Learning to adapt to changing environments in evolving neural networks. Technical Report 95-15, Institute of Psychology, National Research Council, Rome, Italy. [cited at p. 15]

[14] E. Ruppin. Evolutionary autonomous agents: A neuroscience perspective, 2002. Nature Reviews Neuroscience, 3:132-141, 2002. [cited at p. 15]

[15] S. Munroe and A. Cangelosi. Learning and the evolution of language: the role of cultural variation and learning costs in the baldwin effect. *Artificial Life*, 8(4):311–339, 2002. [cited at p. 15]

[16] P.M. Todd and G.F. Miller. Exploring adaptive agency ii: simulating the evolution of associative learning. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 306–315, Cambridge, MA, USA, 1990. MIT Press. [cited at p. 15]

[17] J.J. Grefenstette, C. L. Ramsey, and A. Schultz. Learning sequential decision rules using simulation models and competition. *Machine Learning*, 5:355–381, 1990. [cited at p. 21, 25]

[18] J.J. Bryson. Hierarchy and sequence vs. full parallelism in action selection. In D. Ballin, editor, *Proceedings of the Second Workshop on Intelligent Virtual Agents (IVA '99)*, pages 113–125, Salford, UK, September 1999. [cited at p. 21]

[19] J. K. Rosenblatt and D. W. Payton. A fine-grained alternative to the subsumption architecture for mobile robot control. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 2, pages 317–324, Washington, DC, 1989. IEEE Press. [cited at p. 21, 22]

[20] D. Terzopoulos, T. Rabie, and R. Grzeszczuk. Perception and learning in artificial animals. In *Artificial Life V: Proceedings Fifth International Conference on the Synthesis and Simulation of Living Systems*, pages 346–353, 1996. [cited at p. 21, 22]

[21] S.C. Gadanho and L. Cust. Learning behavior-selection in a multi-goal robot task. In *In NAISO ICAIS proceedings*. ICSC Academic Press, 2002. [cited at p. 21, 22]

[22] S.C. Gadanho and J. Hallam. Robot learning driven by emotions. *Adaptive Behavior.*, 9(1):42–64, 2001. [cited at p. 21]

[23] M. Dorigo and U. Schnepf. Genetics-based machine learning and behaviour based robotics: A new synthesis. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(1):141–154, 1993. [cited at p. 21, 22]

[24] Mataric M.J. Learning in behavior-based multi-robot systems: policies, models, and other agents. *Cognitive Systems Research*, 2:81–93(13), April 2001. [cited at p. 21]

[25] J.J. Bryson. The behavior-oriented design of modular agent intelligence. In R. Kowalszyk, J.P. Müller, H. Tianfield, and R. Unland, editors, *Agent Technologies, Infrastructures, Tools, and Applications for e-Services*, pages 61–76. Springer, 2003. [cited at p. 21, 24]

[26] P. Sengers. Do the thing right: an architecture for action-expression. In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, pages 24–31. ACM Press, 1998. [cited at p. 22]

Appendices

Appendix A

Concepts

Concept	Description
AGENT	The properties of an agent
PLANT	The properties of a plant
PLANT_TYPE_A	The specific properties of a type A plant
PLANT_TYPE_B	The specific properties of a type B plant
MALE	The properties that define a male agent
FEMALE	The properties that define a female agent
FAR	The distance to an object that is considered far away
NEAR	The distance to an object that is considered nearby
REACHABLE	The distance to an object that is within reach
FRONT	The relative position of an object that is considered in front of the agent
FRONT_LEFT	The relative position of an object that is considered front left
FRONT_RIGHT	The relative position of an object that is considered front right
ENERGY_HIGH	The amount of energy that is considered as high
ENERGY_MEDIUM	The amount of energy that is considered as medium
ENERGY_LOW	The amount of energy that is considered as low

Table A.1: Concepts used in the current system

Appendix B

Example Decision Q-Tree

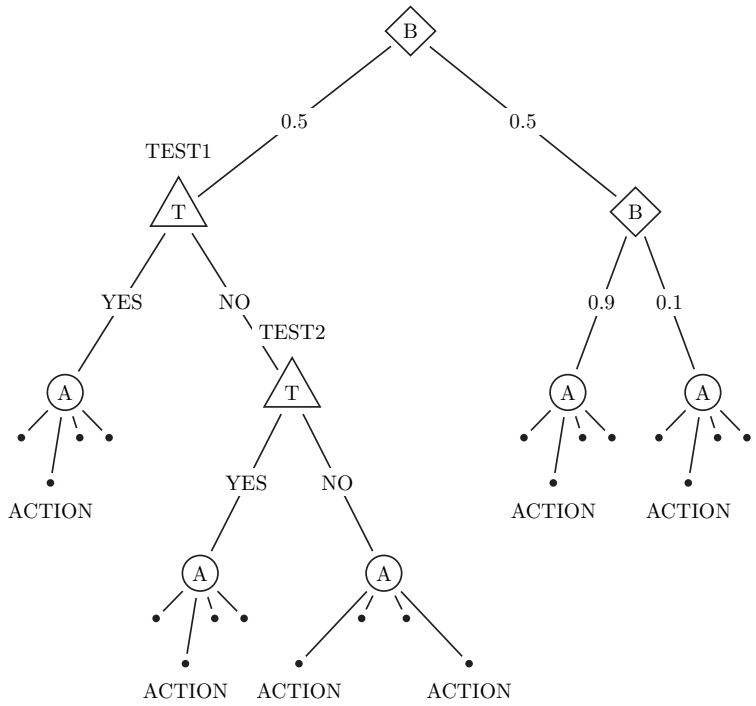


Figure B.1: Example: A valid Decision Q-Tree

Appendix C

Pseudocode

C.1 Decision Trees with Target Set Intersection

```
selected_objects := CIM
node := root_node
while (node != action_node) do
  if (node == test_node)
    if(objects_satisfying_test(CIM).length > 0)

      selected_objects := selected_objects ∩ objects_satisfying_test(CIM)

      node = node.YesNode
    else
      node = node.NoNode
    end if
  else
    ...
  ends
end
```

C.2 Attention Oriented Tree Traversal

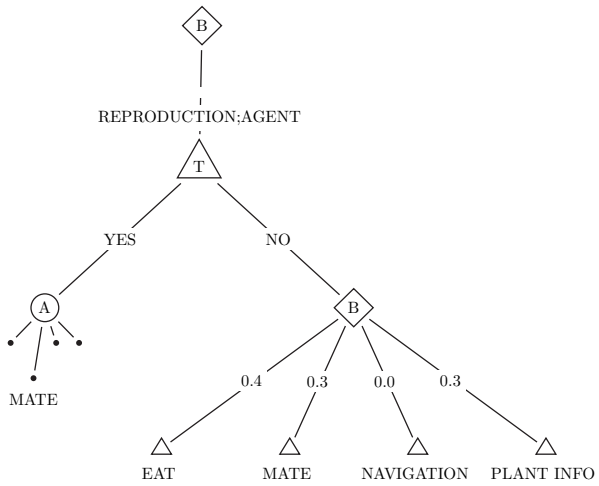
```
selected_objects := CIM
node := root_node
while (node != action_node) do
  if (node == test_node)
    if(objects_satisfying_test(selected_objects).length > 0)

      selected_objects := selected_objects ∩ objects_satisfying_test(selected_objects)

      node = node.YesNode
    else
      node = node.NoNode
```

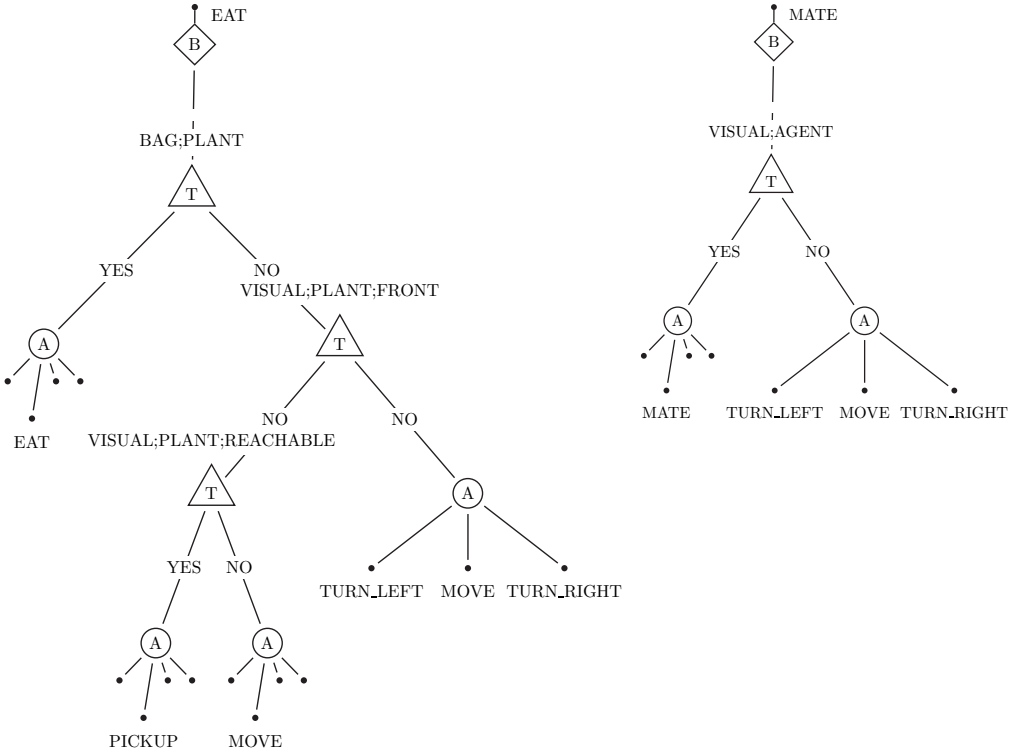
```
    end if
  else
    ...
  end
end
```

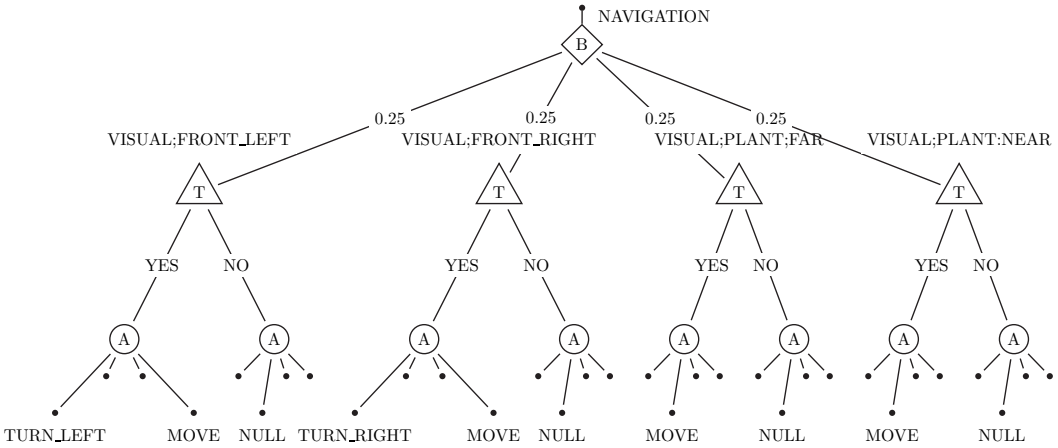
Appendix D

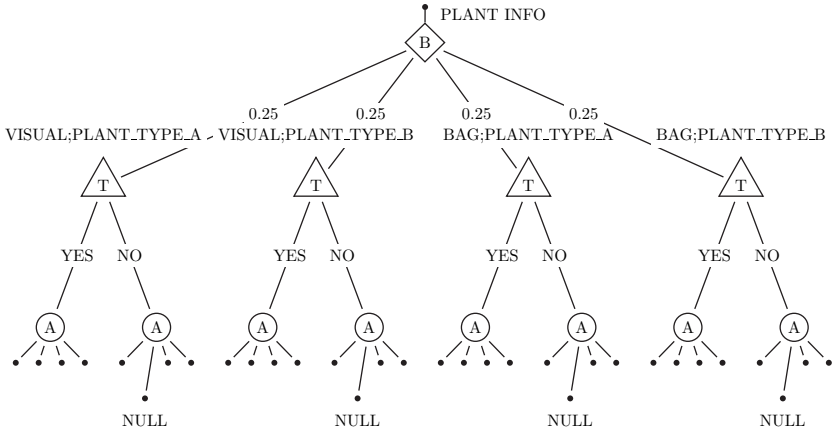
Challenges

D.1 Target Selection

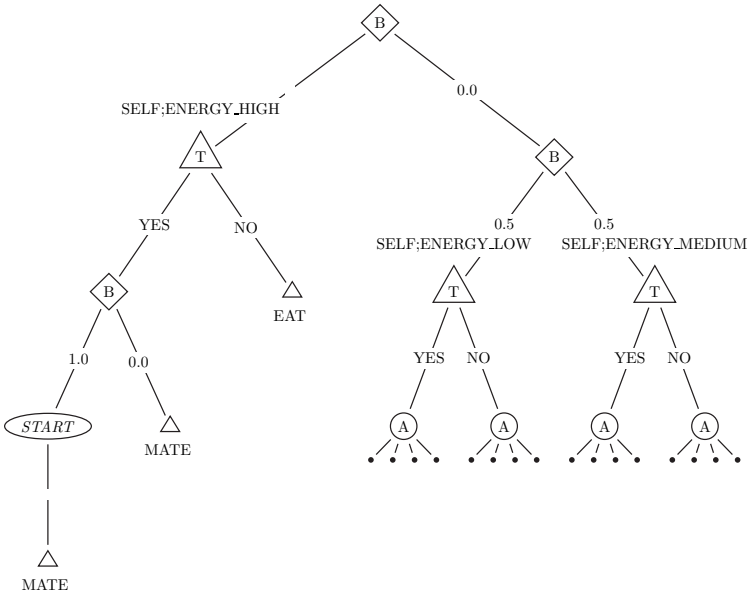


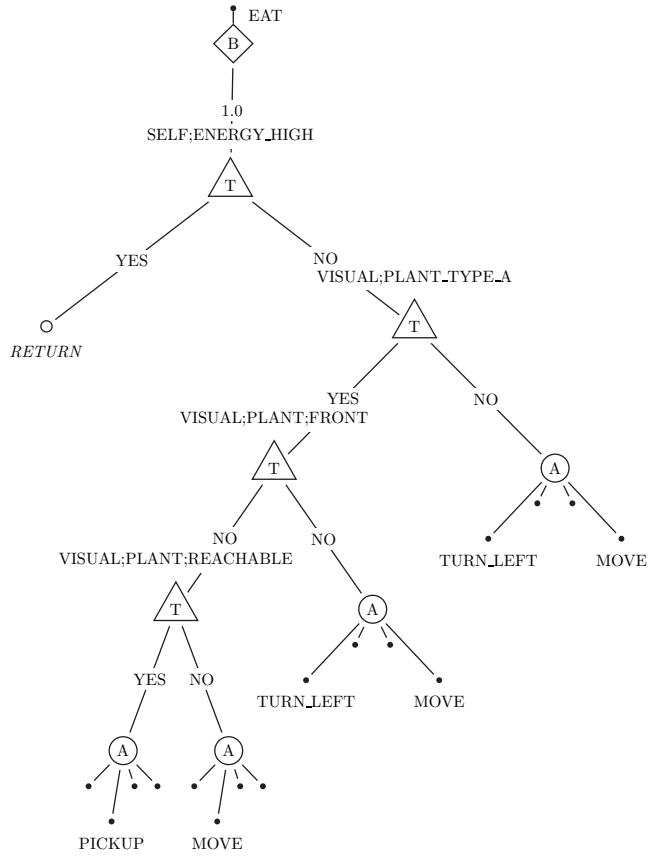

EAT
B
BAG;PLANT
T
YES
NO
A
EAT
VISUAL;PLANT;FRONT
T
NO
NO
VISUAL;PLANT;REACHABLE
T
A
YES
NO
TURN_LEFT
MOVE
TURN_RIGHT
A
A
PICKUP
MOVE
MATE
B
VISUAL;AGENT
T
YES
NO
A
A
MATE
TURN_LEFT
MOVE
TURN_RIGHT

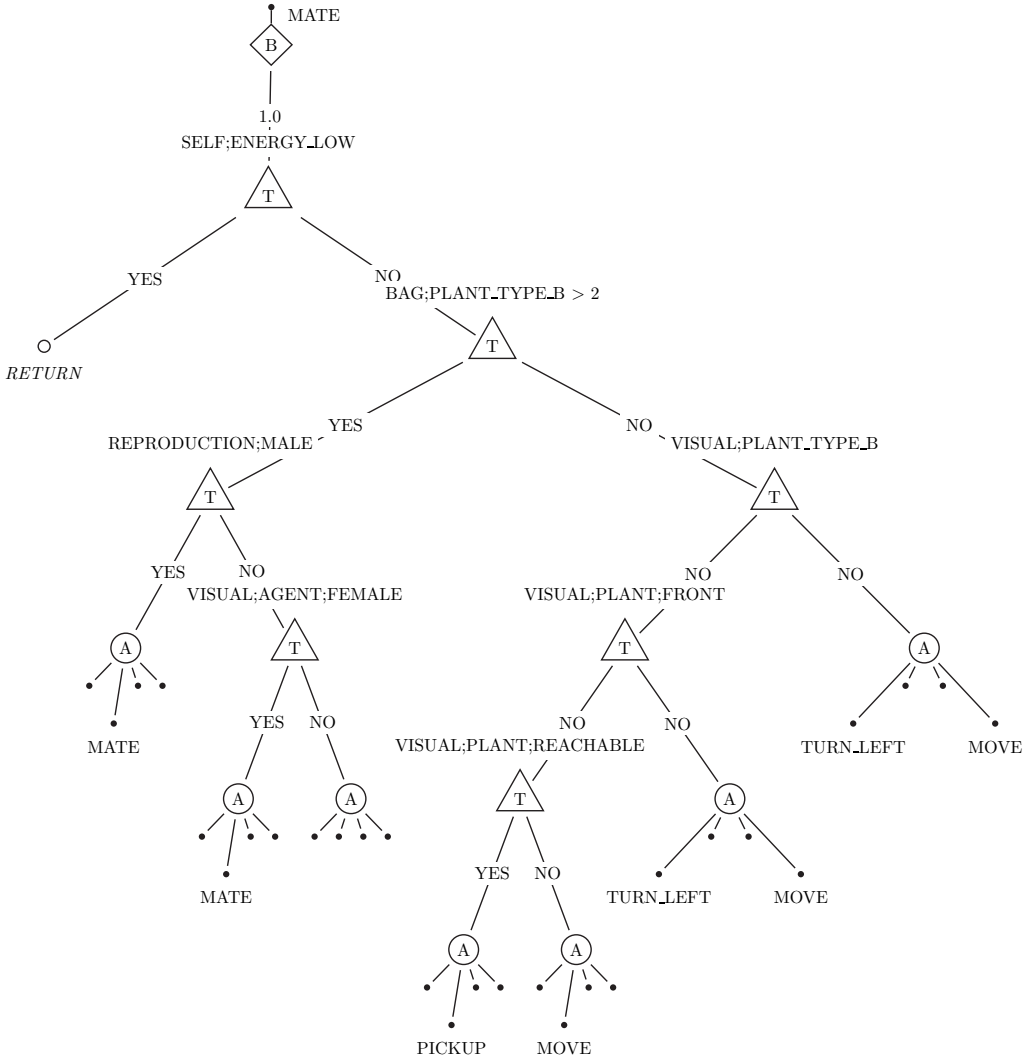

NAVIGATION
B
0.25
0.25
0.25
0.25
VISUAL;FRONT_LEFT
VISUAL;FRONT_RIGHT
VISUAL;PLANT;FAR
VISUAL;PLANT:NEAR
T
T
T
T
YES
NO
YES
NO
YES
NO
YES
NO
A
A
A
A
A
A
A
A
A
A
A
A
TURN_LEFT
MOVE
NULL
TURN_RIGHT
MOVE
NULL
MOVE
NULL
MOVE
NULL


PLANT INFO
B
0.25
0.25
0.25
0.25
VISUAL;PLANT_TYPE_A
VISUAL;PLANT_TYPE_B
BAG;PLANT_TYPE_A
BAG;PLANT_TYPE_B
T
T
T
T
YES
NO
YES
NO
YES
NO
YES
NO
A
A
A
A
A
A
A
A
A
A
A
A
NULL
NULL
NULL
NULL

D.2 Multi-Objective Controllers






MATE
B
1.0
SELF;ENERGY_LOW
T
YES
NO
RETURN
BAG;PLANT_TYPE_B > 2
T
YES
NO
REPRODUCTION;MALE
VISUAL;PLANT_TYPE_B
T
T
YES
NO
NO
NO
VISUAL;AGENT;FEMALE
VISUAL;PLANT;FRONT
A
T
T
A
MATE
YES
NO
NO
NO
TURN_LEFT
MOVE
VISUAL;PLANT;REACHABLE
A
A
T
A
MATE
YES
NO
TURN_LEFT
MOVE
A
A
PICKUP
MOVE

Appendix E

Experiments

E.1 Target Selection

Success	Environment					Agent					DQT			
	Size	# Edible food	# Poisonous Food	Edible energy	Poisonous energy	Visitor	Concepts	Mating age	Weight	Max Energy	Size	Mate Priority	Eat Priority	Pickup Priority
yes	400x400	16000	16000	50000	-75000	A	1	365	70	900000	medium	1	2	3
yes	400x400	16000	16000	50000	-75000	B	1	365	70	900000	medium	1	2	3
no	400x400	16000	16000	50000	-75000	A	1	365	70	900000	small	1	2	3
no	400x400	16000	16000	50000	-75000	B	1	365	70	900000	small	1	2	3
no	400x400	16000	16000	50000	-75000	A	2	365	70	900000	large	1	2	3
no	400x400	16000	16000	50000	-75000	B	2	365	70	900000	large	1	2	3
no	400x400	16000	16000	50000	-75000	A	1	365	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-75000	B	1	365	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-75000	A	1	365	70	900000	medium	3	2	2
no	400x400	16000	16000	50000	-75000	B	1	365	70	900000	medium	3	2	1
no	400x400	16000	16000	50000	-100000	A	1	365	70	900000	medium	3	2	2
no	400x400	16000	16000	50000	-100000	B	1	365	70	900000	medium	3	2	1
no	400x400	16000	16000	50000	-100000	A	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-100000	B	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-50000	A	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-50000	B	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-63000	A	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-63000	B	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-75000	A	1	0	70	900000	medium	2	1	3
no	400x400	16000	16000	50000	-75000	B	1	0	70	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	A	1	0	70	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	B	1	0	70	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	A	1	365	70	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	B	1	365	70	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	A	1	365	300	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	B	1	365	300	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	A	1	365	0	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	B	1	365	0	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	A	1	365	50	900000	medium	2	1	3
no	200x200	16000	16000	50000	-75000	B	1	365	50	900000	medium	2	1	3
no	200x200	16000	16000	20000	-20000	A	1	365	50	20000	medium	2	1	3
no	200x200	16000	16000	20000	-20000	B	1	365	50	20000	medium	2	1	3
no	200x200	16000	16000	50000	-40000	A	1	365	50	20000	medium	2	1	3
no	200x200	16000	16000	20000	-40000	B	1	365	50	20000	medium	2	1	3
no	200x200	16000	16000	400	-800	A	1	365	10	20000	medium	2	1	3
no	200x200	16000	16000	400	-800	B	1	365	10	20000	medium	2	1	3
no	200x200	16000	16000	800	-1600	A	1	365	10	20000	medium	2	1	3
no	200x200	16000	16000	800	-1600	B	1	365	10	20000	medium	2	1	3
no	200x200	10000	10000	400	-800	A	1	365	10	20000	medium	2	1	3
no	200x200	10000	10000	400	-800	B	1	365	10	20000	medium	2	1	3
no	200x200	10000	10000	800	-1600	A	1	365	10	20000	medium	2	1	3
no	200x200	10000	10000	800	-1600	B	1	365	10	20000	medium	2	1	3
no	200x200	* 10000	10000	800	-1600	A	1	365	10	20000	medium	2	1	3
no	200x200	* 10000	10000	800	-1600	B	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	400	-800	A	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	400	-800	B	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	800	-1600	A	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	800	-1600	B	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	400	-800	A	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	400	-800	B	1	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	800	-1600	A	3	365	10	20000	medium	2	1	3
no	200x200	* 5000	5000	800	-1600	B	3	365	10	20000	medium	2	1	3
no	100x100	* 2000	2000	800	-1600	C	3	365	10	20000	medium	2	1	3
no	100x100	* 2000	2000	800	-1600	B	3	365	10	20000	medium	2	1	3
no	100x100	* 2000	2000	800	-1600	C	3	365	10	20000	medium	2	1	3

Visitors: A = TSI, B = TSI + AOTP, C = no target set filtering and ordinary tree traversal
Concepts: 1 = random sized correct set, 2 = large correct set, 3 = set with incorrect concepts
Edible & Poisonous food: (*) with NEW TIES growth distance 1, else distance 10

Table E.1: Target Selection Experiments

E.2 Multi-Objective Controllers

Success	Environment						Agent			DQT			
	Landscape	# Edible food	# Mating plants	Edible energy	Mating plants needed	Plant Growth	Behaviour Switch	Mating age	Max Age	Max Energy	Size	Mate Priority	Eat Priority	Collect Priority
yes	150x150 (1)	6000	400	1000	2	1	yes	0	∞	25000	medium	1	2	3
yes	150x150 (1)	6000	400	1000	2	1	no	0	∞	25000	medium	1	2	3
no	150x150 (1)	6000	400	1000	2	1	yes	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	400	1000	2	1	no	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	400	1000	2	1	yes	0	7300	25000	small	1	2	3
no	150x150 (1)	6000	400	1000	2	1	no	0	7300	25000	small	1	2	3
no	150x150 (1)	6000	400	1000	2	1	yes	0	7300	25000	large	1	2	3
no	150x150 (1)	6000	400	1000	2	1	no	0	7300	25000	large	1	2	3
no	150x150 (1)	6000	800	1000	2	1	yes	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	3000	1000	2	1	no	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	3000	1000	2	1	yes	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	3000	1000	3	1	no	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	3000	1000	3	1	yes	0	7300	25000	medium	1	2	3
no	150x150 (1)	6000	3000	1000	3	1	no	0	7300	25000	medium	1	2	3
no	400x400 (1)	15000	5000	1000	3	2	yes	0	7300	25000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	no	0	7300	25000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	yes	0	7300	25000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	no	0	7300	25000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	yes	0	7300	50000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	no	0	7300	50000	medium	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	yes	0	7300	25000	medium	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	no	0	7300	25000	medium	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	yes	0	7300	50000	medium	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	no	0	7300	50000	medium	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	yes	0	7300	25000	medium	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	no	0	7300	25000	medium	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	yes	0	7300	50000	medium	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	no	0	7300	50000	medium	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	yes	0	7300	25000	large	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	no	0	7300	25000	large	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	yes	0	7300	25000	large	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	no	0	7300	25000	large	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	yes	0	7300	25000	large	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	no	0	7300	25000	large	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	yes	0	7300	25000	small	2	1	3
no	400x400 (1)	15000	5000	1000	3	2	no	0	7300	25000	small	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	yes	0	7300	25000	small	2	1	3
no	400x400 (2)	15000	5000	1000	3	1	no	0	7300	25000	small	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	yes	0	7300	25000	small	2	1	3
no	400x400 (3)	15000	5000	1000	3	1	no	0	7300	25000	small	2	1	3
no	400x400 (1)	5000	5000	1000	3	2	yes	0	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	1000	3	2	no	0	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	1000	3	1	yes	0	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	1000	3	1	no	0	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	1000	3	1	yes	0	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	1000	3	1	no	0	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	1000	3	2	yes	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	1000	3	2	no	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	1000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	1000	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	1000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	1000	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	500	3	2	yes	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	500	3	2	no	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	500	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	500	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	500	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	500	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	4000	3	2	yes	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	4000	3	2	no	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	4000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	4000	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	4000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	4000	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	16000	3	2	yes	365	7300	50000	medium	2	1	3
no	400x400 (1)	5000	5000	16000	3	2	no	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	16000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (2)	5000	5000	16000	3	1	no	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	16000	3	1	yes	365	7300	50000	medium	2	1	3
no	400x400 (3)	5000	5000	16000	3	1	no	365	7300	50000	medium	2	1	3

Landscape: 1 = randomly distributed, 2 = fully separated, 3 = some overlap
Plant Growth: 1 = Food with NEW TIES growth and Mating plants with SugarScape growth, 2 = Both with NEW TIES growth

Table E.2: Multi-Objective Controller Experiments

List of Figures

List of Tables