Санкт-Петербургский государственный университет информационных технологий, механики и оптики
Факультет информационных технологий и программирования
Кафедра «Компьютерные Технологии»

Отчет по лабораторной работе

«Построение управляющих автоматов с помощью генетических алгоритмов»

А.К. Вихарев

16 октября 2010 г.

Оглавление

1. Введение 3
2. Постановка задачи 3
3. Алгоритм 3
   - 3.1. Представление особи 3
   - 3.2. Тип генетического алгоритма 3
   - 3.3. Способ отбора 4
4. Полученные результаты 4
   - 4.1. Сравнение функций рангового отбора 4
   - 4.2. Сравнение моделей генетического алгоритма 4
   - 4.3. Сравнение представлений особи 4
5. Выводы 5

1. Введение

В лабораторной работе требуется реализовать генетический алгоритм построения автомата, решающего задачу об «Умном муравье-3». Для этого реализуется набор плагинов для виртуальной лаборатории на языке Java [1], который реализует конкретный тип генетического алгоритма и конкретное представление особей-автоматов. При подготовке к лабораторной работе использованы материалы [3],[4].

2. Постановка задачи

В задаче об «Умном муравье-3» рассматривается поле, располагающееся на поверхности тора и имеющее размер 32 на 32 клетки. Каждая клетка поля с некоторой, заранее определенной, вероятностью содержит яблоко. Муравей видит восемь клеток перед собой и может выполнять одно из следующих действий:

- повернуть направо
- повернуть налево
- сделать шаг вперед, и если в новой клетке есть яблоко, то съесть его

Максимальное число шагов – 200. Цель задачи – создать муравья, который управляется автоматом с фиксированным числом состояний и съедает максимальное число яблок.

3. Алгоритм

3.1. Представление особи

Особью генетического алгоритма для решения данной задачи является конечный автомат Мили – детерминированный конечный автомат, каждому переходу которого сопоставлено некоторое действие (в данной задаче это либо шаг вперед, либо поворот вправо или влево).

Входным алфавитом этого автомата являются значения восьми логических переменных, обозначающих наличие или отсутствие еды в каждой из видимых муравьем клеток. Состояния в автомате хранятся в виде дерева решений [2]. В вершинах деревьев находятся входные переменные, в листьях дерева хранятся номера состояний, в которые осуществляется переход, и действие, совершаемое при этом муравьем. Число состояний автомата является настраиваемым параметром алгоритма. Все генерируемые алгоритмом автоматы имеют одинаковое число состояний, не изменяющееся во время выполнения алгоритма.

3.2. Тип генетического алгоритма

В работе использован клеточный генетический алгоритм [5]. Клеточный генетический алгоритм отличается от канонического тем, что популяция представляет собой таблицу из $n \times m$ клеток, в каждой клетке которой находится одна особь, но также существуют вариации алгоритма, в которых в каждой клетке таблицы находится популяция особей. В данной работе использована версия алгоритма, в которой в каждой клетке находится одна особь. На каждом шаге алгоритма особь в каждой клетке скрещивается с особями в соседних клетках, и из получившегося набора особей выбирается одна, которая помещается на место старой особи в клетке. Новая особь может с некоторой вероятностью мутировать.

3.3. Способ отбора

В работе использован ранговый способ отбора. В этом способе набор особей, из которых производится выборка, сортируется по возрастанию значения функции приспособленности, затем каждой особи, в зависимости от положения в отсортированном списке, присваивается ранг. Вероятность для i-ой особи пройти отбор равна $\frac{f(r_i)}{\sum\limits_{i=1}^{n} f(r_i)}$, где r_i – ранг i-ой особи, f – функция, применяемая к рангу. В данной работе в качестве f были использованы функции x^n, e^x.

4. Полученные результаты

4.1. Сравнение функций рангового отбора

Сравнивались различные варианты функции f, применяемой в ранговом отборе. Параметры алгоритма были следующие: муравей тестировался на трех заданных картах, у каждой особи было три состояния, размер поля популяции 10 на 10, вероятность мутации каждой особи 0.05. Алгоритм останавливался при достижении 1000 поколений. Рассмотренные варианты функции f: x, x^2, x^3, x^5, e^x.

Функция $f = x$ показала плохие результаты, среднее значение функции приспособленности в популяции не поднималось выше 10. Функция $f = x^2$ показала средний результат функции приспособленности 60, наивысший 100. Результат был достигнут к 500 поколению. Функция $f = x^3$ показала средний результат функции приспособленности 83, наивысший 103. Результат был достигнут к 350 поколению. Функция $f = x^5$ показала средний результат функции приспособленности 76, наивысший 95. Результат был достигнут к 100 поколению. Функция $f = e^x$ показала средний результат функции приспособленности 83, наивысший 96, результат был достигнут к 150 поколению. Результирующие графики представлены в приложении (рис. 1–5).

4.2. Сравнение моделей генетического алгоритма

Сравнивалась каноническая модель генетического алгоритма с клеточной моделью. Параметры алгоритма были следующие: муравей тестировался на трех заданных картах, у каждой особи было три состояния. В клеточной модели размер поля популяции 10 на 10, вероятность мутации каждой особи 0.05. В канонической модели использовались настройки по умолчанию. В качестве способа отбора был взят ранговый отбор. Функцией от ранга была $f = x^{2.5}$. Алгоритм с клеточной моделью останавливался через 1000 поколений, алгоритм с канонической моделью через 10000.

Среднее значение функции приспособленности в канонической модели было равно 73, наивысшее 97, достигнуто было на 6900 поколении. Среднее значение функции приспособленности клеточной модели было равно 70, наивысшее значение было равно 100, достигнуто было на 425 поколении. Результирующие графики представлены в приложении (рис. 6,7).

4.3. Сравнение представлений особи

Сравнивались представления особи в виде деревьев решений и в виде полных таблиц переходов. Параметры алгоритма были следующие: муравей тестировался на трех заданных картах, у каждой особи было три состояния. Использовалась клеточная модель генетического алгоритма. Размер поля популяции 10 на 10, вероятность мутации каждой особи 0.05. В качестве способа отбора был взят ранговый отбор. Функцией от ранга была $f = x^{2.5}$. Алгоритм останавливался при достижении 1000 поколений.

Среднее значение функции приспособленности при использовании представления особи в виде полных таблиц переходов было равно 88, максимальное значение достигло 96 на 150 поколении. При использовании представления особи в виде деревьев решений среднее значение функции приспособленности было равно 65, максимальное значение достигло 99 на поколении 275. Результирующие графики представлены в приложении (рис. 8–9).

5. Выводы

Исследование различных вариантов рангового отбора показало, что для клеточной модели генетического алгоритма в данной задаче следует выбирать функцию от ранга $f = x^n$, где $2 < n < 3$. При таком способе отбора обеспечивается равномерная сходимость генетического алгоритма. При $n > 3$ алгоритм сходится слишком быстро, что способствует попаданию в локальный максимум и сужает диапазон поиска решений. При $n < 2$ сходимость алгоритма плохая, в результате чего алгоритм игнорирует локальные максимумы функции приспособленности. Наиболее хорошие результаты дает функция $f = x^{2.5}$. Для дальнейших исследований можно предложить ввести в функцию линейную добавку, изменяемую во время работы алгоритма.

Исследование различных моделей генетического алгоритма показало, что клеточная модель дает намного более быстрое схождение алгоритма. К этому могло привести то, что в клеточной модели алгоритма на каждом шаге популяция полностью обновляется, в то время как в канонической модели часть особей переходит из старого поколения в новое. Найденные оптимальные решения при этом отличаются несущественно, что свидетельствует об эффективности обоих алгоритмов.

Исследование различных представлений особи показало, что при выбранных параметрах алгоритма представление в виде полных таблиц обеспечивает более быструю сходимость алгоритма. Среднее по популяции значение функции приспособленности было выше, чем при использовании представления в виде деревьев решений. Но при этом наилучшее решение, найденное алгоритмом, использующим деревья решений, было лучше, чем при использовании полных таблиц. В целом, использование представления автоматов в виде деревьев решений требует намного более тщательного подбора параметров особи, чем другие решения. Но эта сложность компенсируется более вероятным получением лучших результатов.

Литература

[1] Инструкция по работе с виртуальной лабораторией на языке Java.
http://is.ifmo.ru/genalg/labs_2010-2011/interface_manual.pdf

[2] Данилов В.Р., Шалыто А.А. Метод генетического программирования для генерации автоматов, представленных деревьями решений.
http://is.ifmo.ru/download/2008-03-07-danilov.pdf

[3] А. А. Давыдов, Д. О. Соколов, Ф. Н. Царев. Применение генетических алгоритмов для построения автоматов мура и систем взаимодействующих автоматов мили на примере задачи об «умном муравье».
http://is.ifmo.ru/works/_2009_08_12_davydov.pdf

[4] Ф. Н. Царев, А. А. Шалыто. Разработка технологии генетического программирования для генерации автоматов управления системами со сложным поведением.
http://is.ifmo.ru/present/_genetic-itmo.ppt

[5] A. J. Nebro, J. J. Durillo, F. Luna, B. Dorronsoro, and E. Alba A Cellular Genetic Algorithm for Multiobjective Optimization.

Приложение

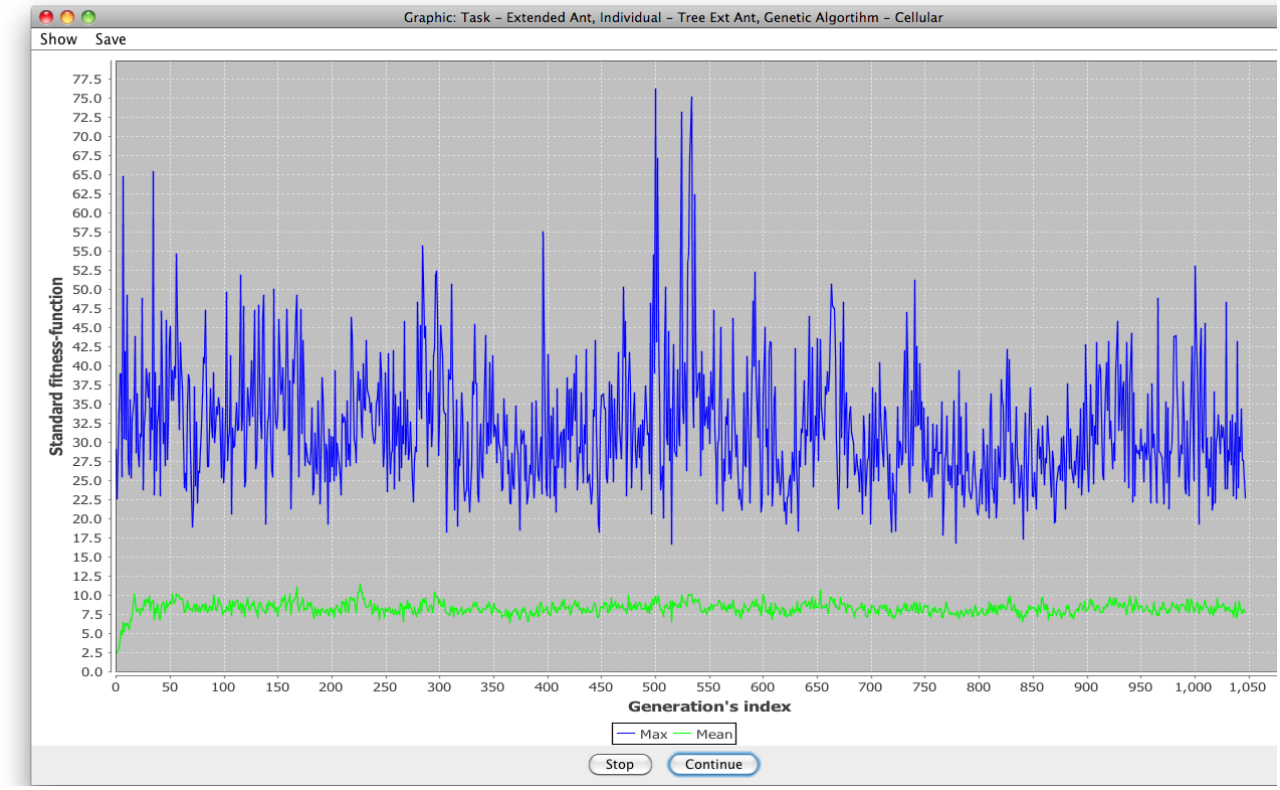

Рис. 1. Алгоритм с функцией $f = x$

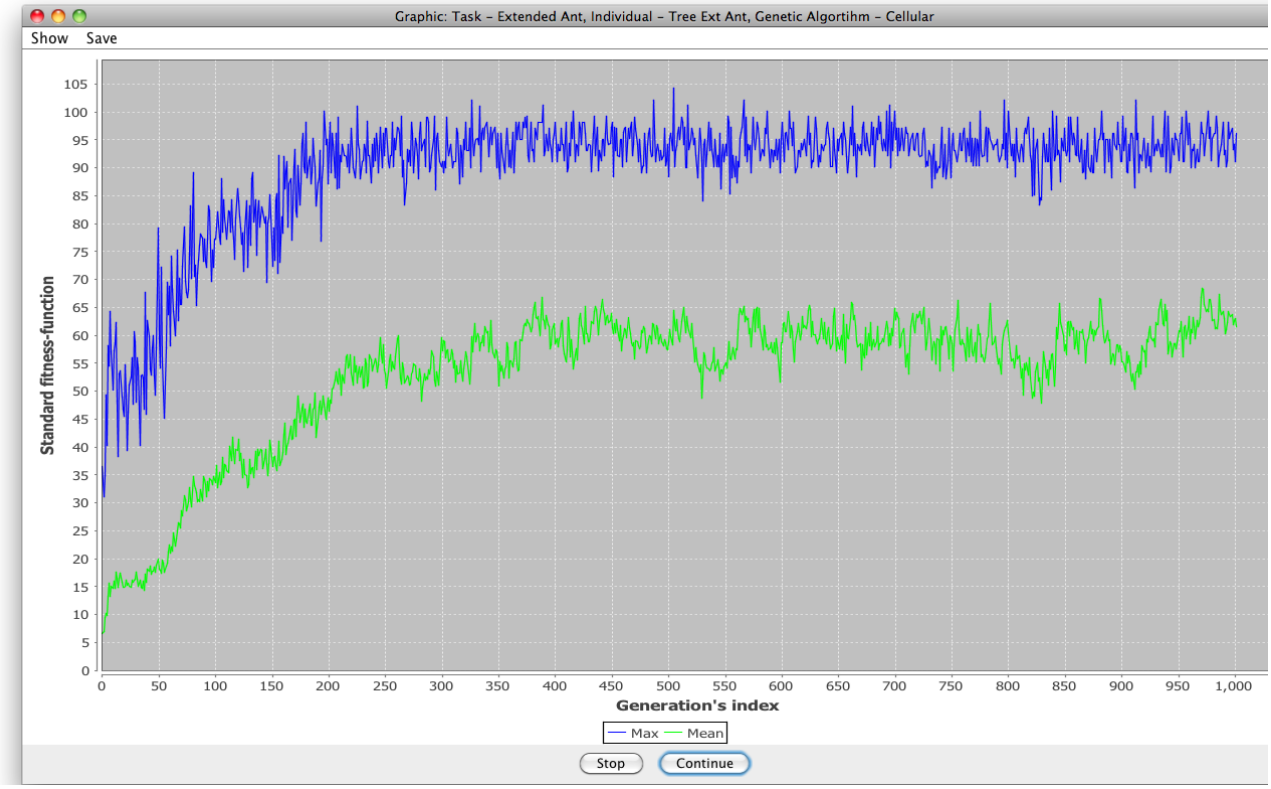

Рис. 2. Алгоритм с функцией $f = x^2$

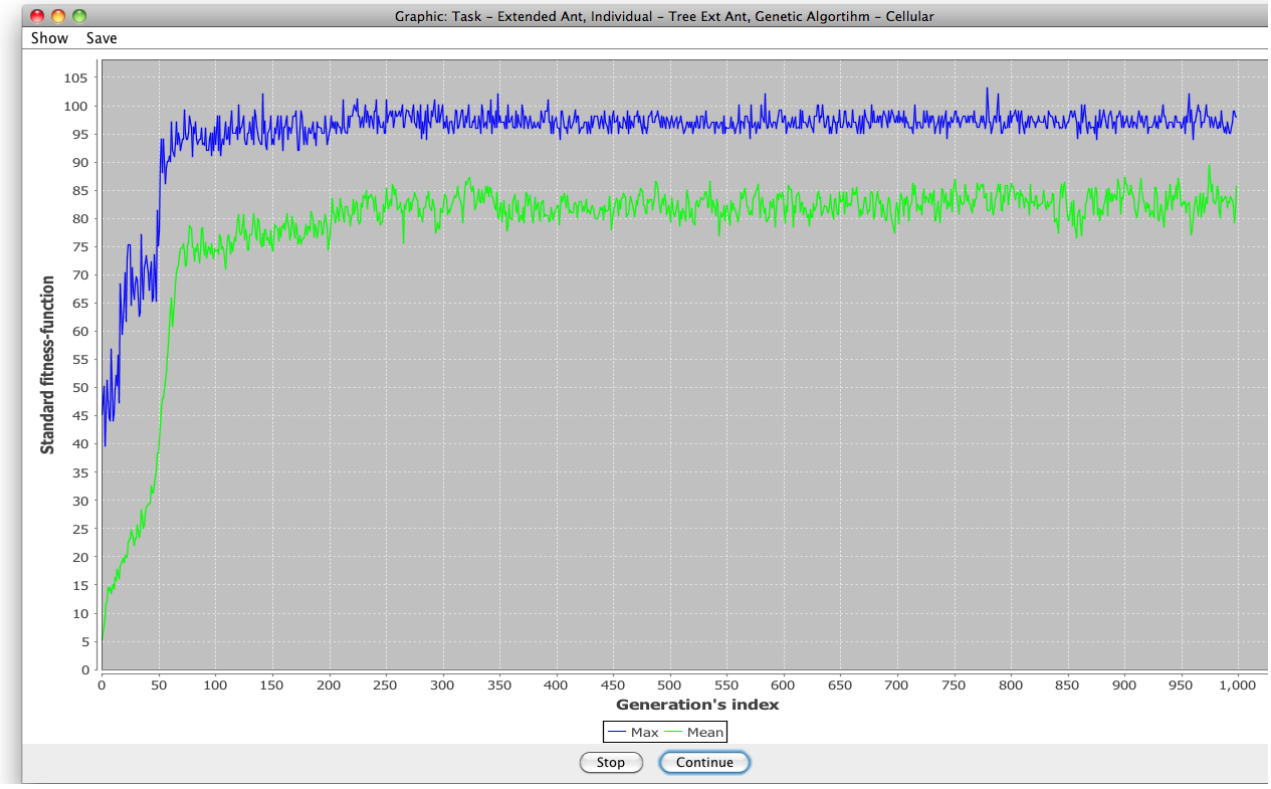

Рис. 3. Алгоритм с функцией $f = x^3$

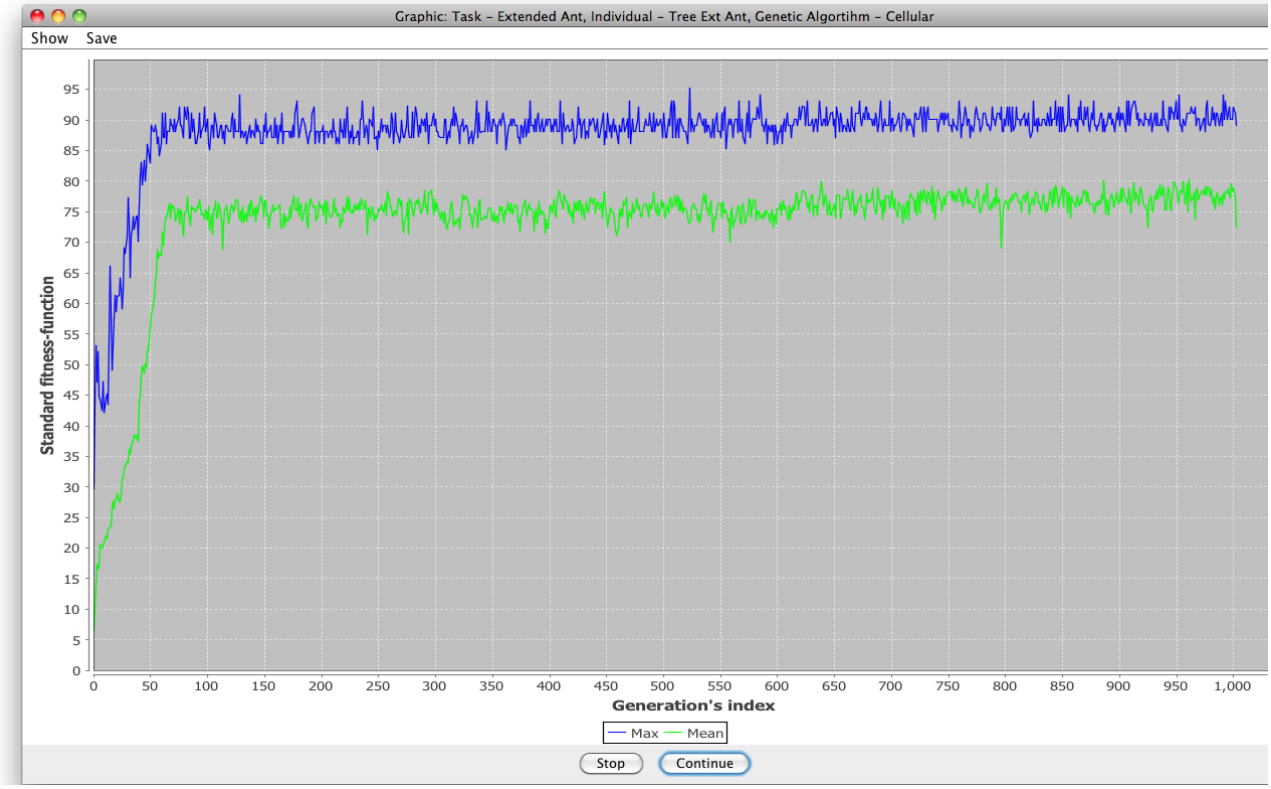

Рис. 4. Алгоритм с функцией $f = x^5$

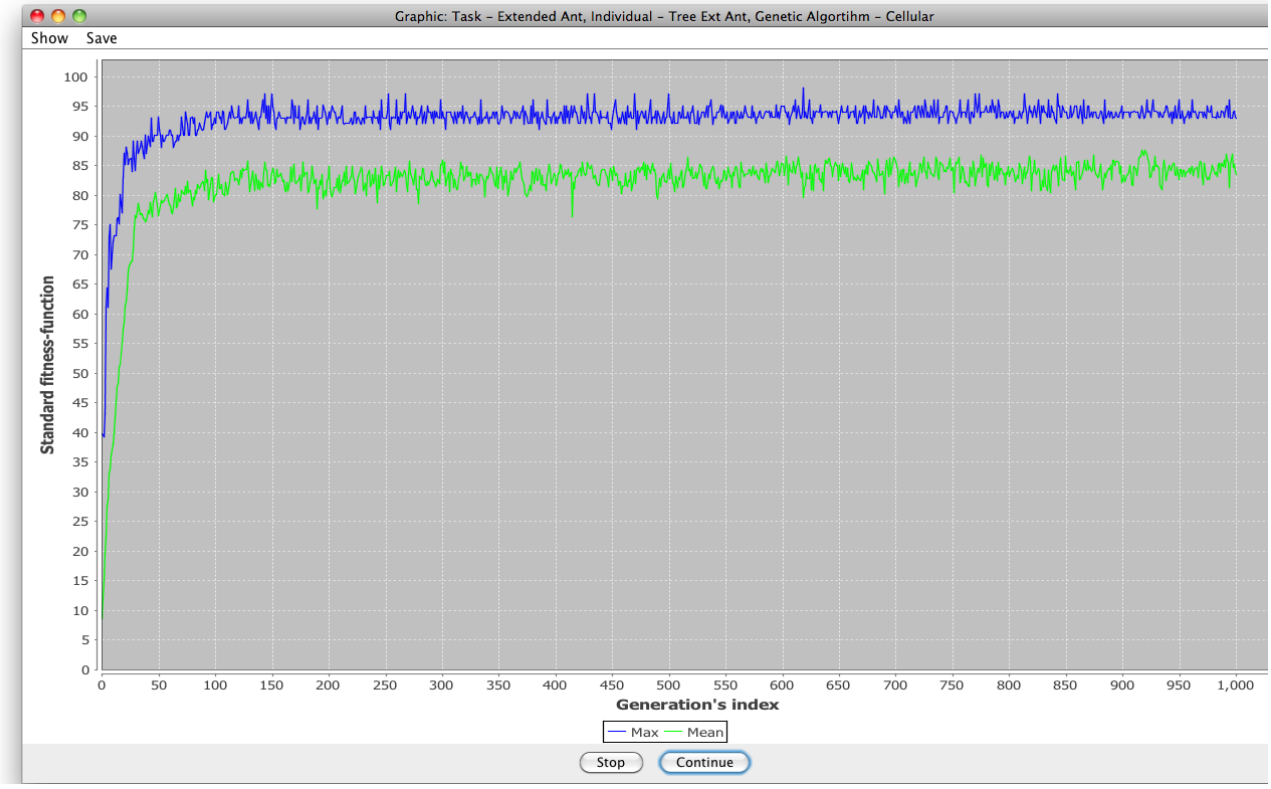

Рис. 5. Алгоритм с функцией $f = e^x$

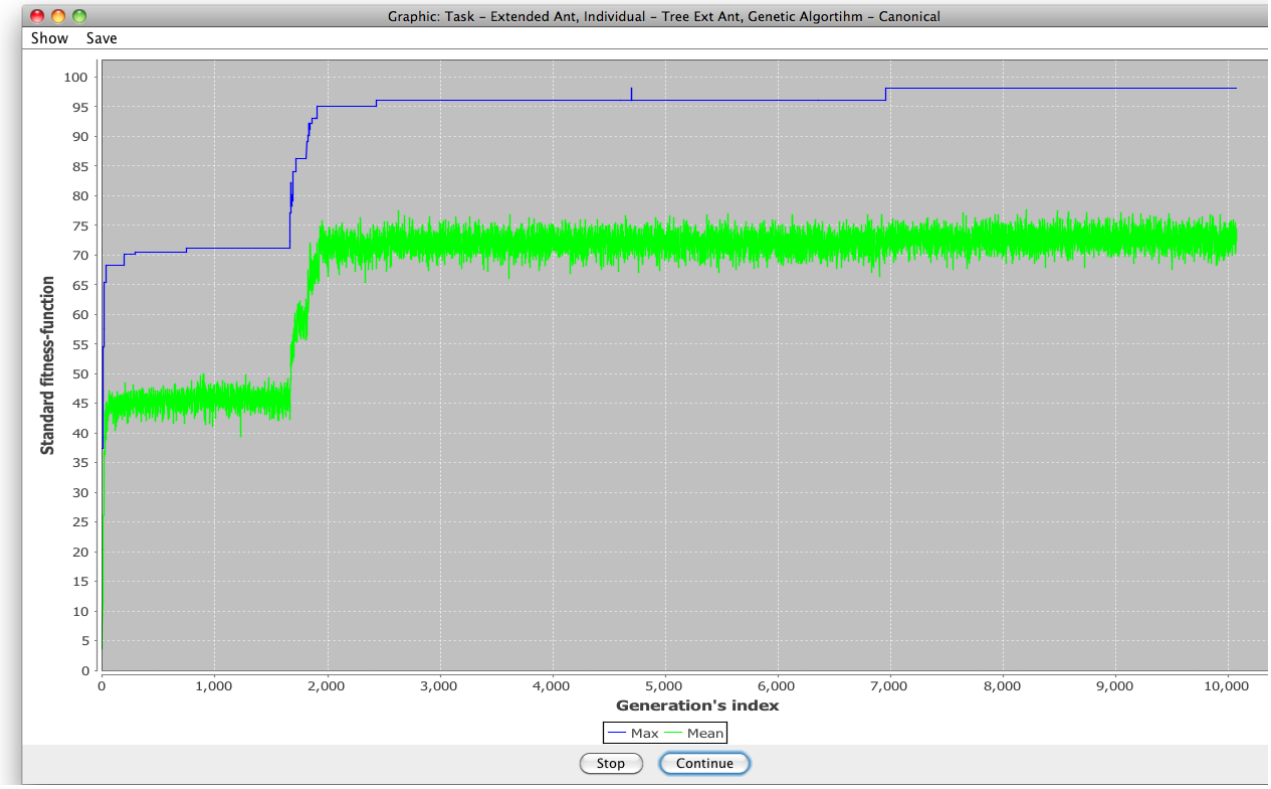

Рис. 6. Канонический генетический алгоритм

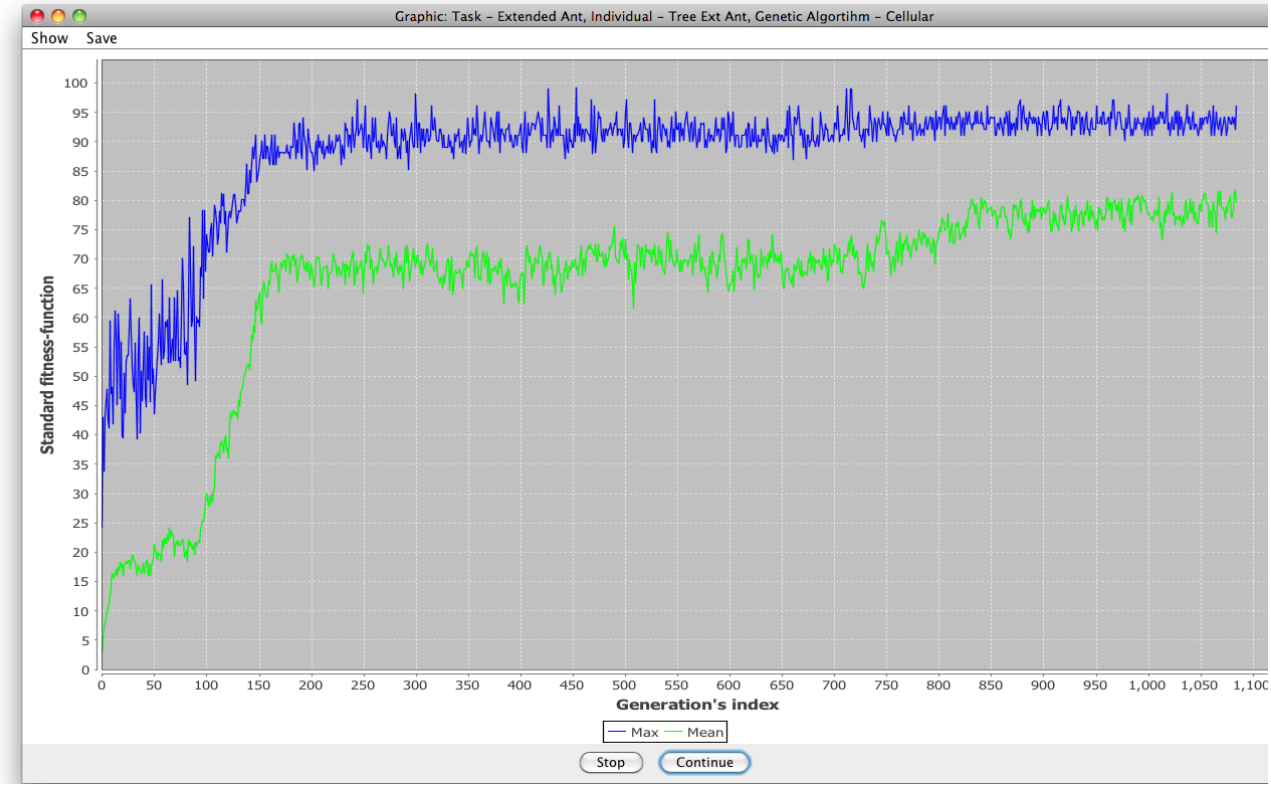

Рис. 7. Клеточный генетический алгоритм

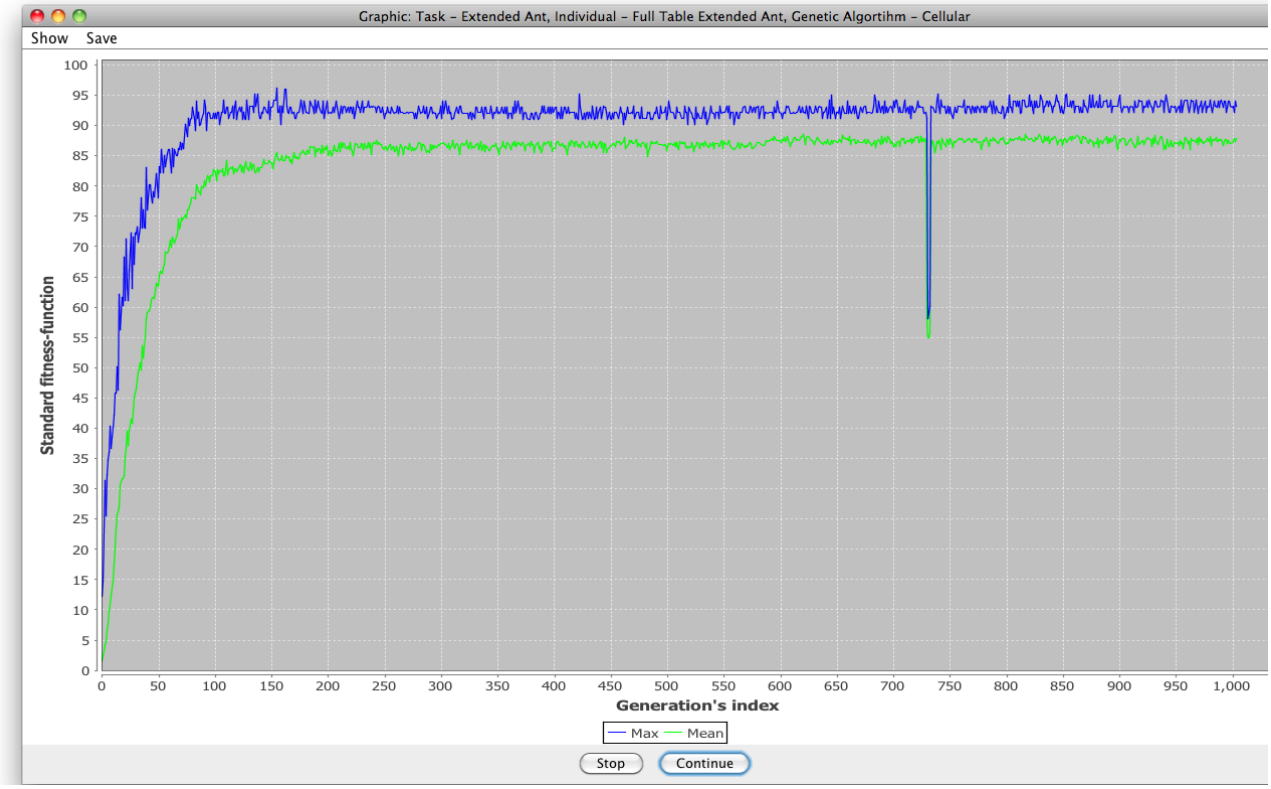

Рис. 8. Представление в виде полных таблиц

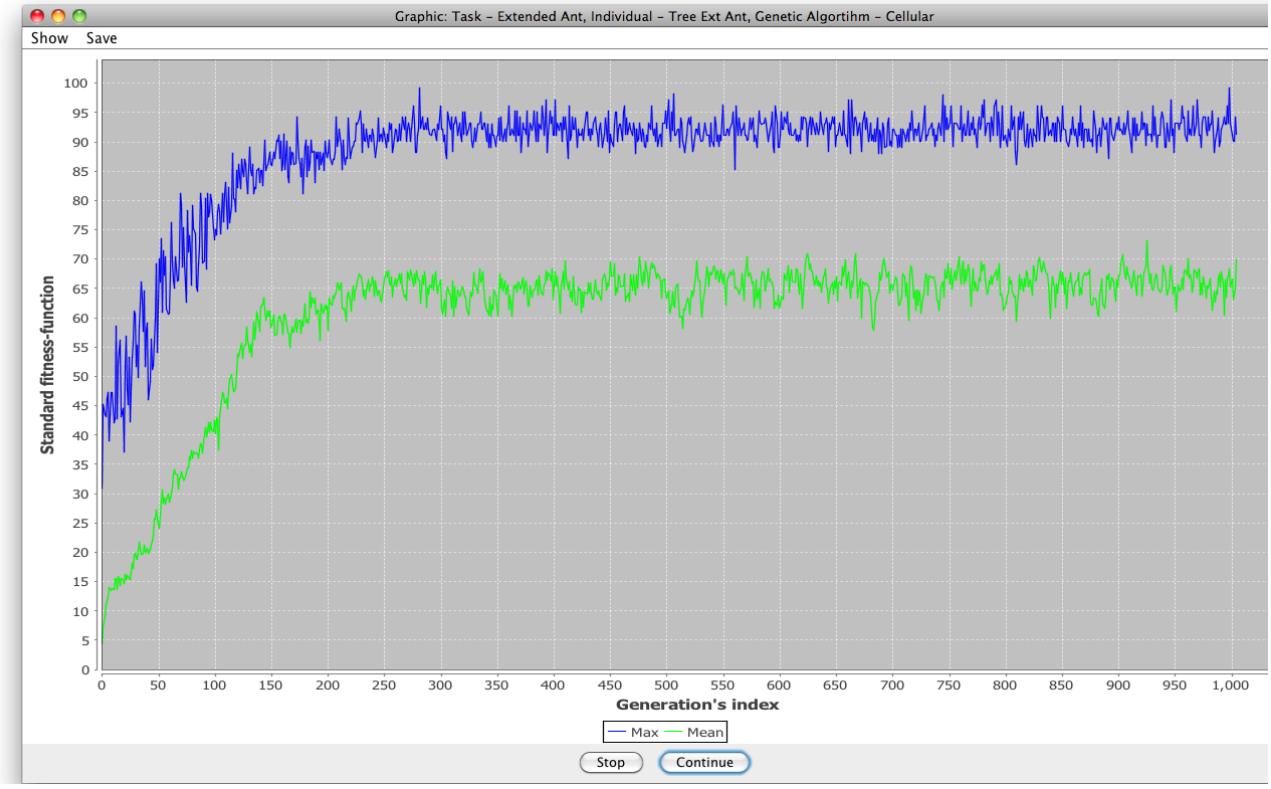

Рис. 9. Представление в виде деревьев решений