Материал опубликован в сборнике докладов III Международной научно-практической конференции «Современные информационные технологии и ИТ-образование». ВМК МГУ. М.: МАКС Пресс, 2008, с. 179 – 183.

Виртуальная лаборатория обучения генетическому программированию для генерации управляющих конечных автоматов

А. А. Давыдов, Д. О. Соколов, Ф. Н. Царев, А. А. Шалыто,
Санкт-Петербургский государственный университет информационных технологий, механики и оптики, sokolov@rain.ifmo.ru

В докладе описывается виртуальная лаборатория, предназначенная для обучения генетическому программированию для построения управляющих конечных автоматов.

1. Введение

Автоматное программирование [1, 2] – парадигма программирования, предложенная в 1991 году в России. При ее использовании программу предлагается строить в виде совокупности поставщиков событий, системы взаимодействующих конечных автоматов и объектов управления. Поставщик событий характеризуется множеством событий, которые он может генерировать. Объект управления характеризуется множеством вычислительных состояний, а также двумя наборами функций: множеством входных переменных, отображающих вычислительное состояние в логическое значение (истина или ложь), и множеством выходных воздействий, позволяющих изменять вычислительное состояние. Каждый автомат имеет конечное множество управляющих состояний, функцию переходов и функцию выходов.

Из изложенного следует, что ядром автоматных программ являются конечные автоматы. Однако во многих случаях эвристическое построение автоматов весьма трудоемко или невозможно. Часто автоматы могут быть построены автоматически, используя генетическое программирование. При генерации автоматов резко повышается уровень автоматизации программирования программ рассматриваемого класса, так как по автоматам код также может быть сгенерирован автоматически. При этом вручную на целевом языке программирования пишутся только функции поставщиков событий и входных и выходных воздействий. Выполненные проекты (http://is.ifmo.ru/unimod-projects/) показали, что

при таком подходе для многих задач управления до70–80% кода строится автоматически.

Из изложенного следует важность обучения генетическому программированию, по крайней мере, для генерации автоматов. Однако эта разновидность эволюционных алгоритмов является весьма сложной для обучения и без проведения вычислительных экспериментов ее сложно освоить.

Поэтому весьма актуальна разработка виртуальной лаборатории обучения генетическому программированию для генерации управляющих конечных автоматов.

2. Основные положения

Виртуальная лаборатория разработана с использованием языка программирования *Java*. Она является мультиплатформенной и имеет модульную архитектуру – содержит ядро, предоставляющее базовую функциональность, которая расширяется за счет подключаемых модулей (плагинов).

Ядро позволяет просматривать и сохранять в виде файлов графики зависимости значений некоторых функций (например, максимального, минимального и среднего значения функции приспособленности особей поколения) от номера поколения. Кроме этого поддерживается возможность визуализации особей и их сохранения в текстовом формате.

В качестве подключаемых модулей выступают:

- решаемые задачи;
- функции, графики которых строятся ядром программы;
- реализации различных алгоритмов генетического программирования;
- реализации различных представлений особей и операций скрещивания и мутации для алгоритмов генетического программирования;
- визуализаторы особей (автоматов) и управляемых ими объектов (они зависят от конкретной задачи и от конкретного представления особи).

При этом отметим, что алгоритмы генетического программирования и особи в некотором смысле независимы – предполагается, что любой алгоритм рассматриваемого класса может работать с любым представлением особи.

В состав лаборатории входят следующие модули:

- задачи для решения: «Умный муравей» [3 – 6] и «Умный муравей-3» [7];
- функции, возвращающее максимальное и среднее значения функции приспособленности особей в поколении;

- классический и островной генетические алгоритмы [8];
- представления хромосомы в виде полных таблиц переходов для одиночных автоматов Мили и Мура, а также для систем автоматов Мили;
- визуализатор, моделирующий работу автомата в задачах «Умный муравей» и «Умный муравей-3».

При необходимости решения задач, в том числе отличных от указанных, набор подключаемых модулей может быть расширен.

На рис. 1 приведена структура виртуальной лаборатории.

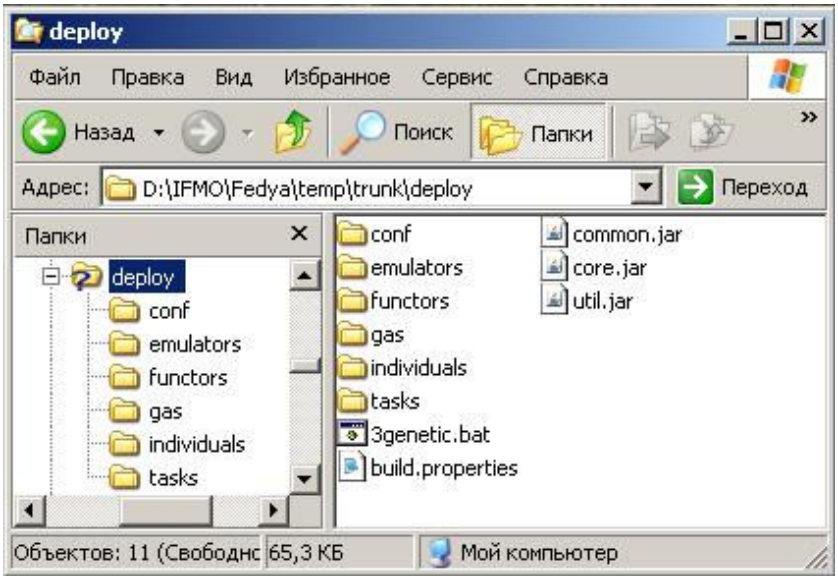

Рис. 1. Структура виртуальной лаборатории

3. Создание встраиваемых модулей

Для того чтобы была возможность собрать встраиваемый модуль для виртуальной лаборатории, необходимо соблюдать ряд правил:

- модуль должен представлять собой jar-архив;
- манифест должен содержать параметр `Main-Class` – указывающий на экземпляр класса, реализующего интерфейс `Loader<?>` из файла `common.jar`. Метод `load` данного интерфейса должен возвращать объект основного класса.

3.1. Создание задачи

Задача должна реализовывать интерфейс `Task`.

```
public interface Task {

    public JPanel getStandardVisualizator(Solution
solution);
    public double standardFitnessFunction(Solution
solution);
    public String getDesciption();
    public String getName();
    public String getShortName();
}
```

Методы:

- `getStandardVisualizator` – возвращает панель с изображением задачи;
- `standardFitnessFunction` – возращает значение значение функции приспособленности по экземпляру решения задачи (этот метод не обязательно использовать при работе алгоритма генетического программирования);
- `getDesciption`, `getName`, `getShortName` – возвращают название и описание задачи.

3.2. Создание особи, алгоритма и функций

Создание встраиваемых модулей особи, алгоритма генетического программирования и функций, отображаемых ядром лаборатории, практически не отличается от создания задачи. Для начала необходимо реализовать соответствующий интерфейс `IndividualFactory`, `GA`, `Functor`.

4. Пример работы виртуальной лаборатории при решении задачи «Умный муравей-3»

На рис. 2 приведен пример работы алгоритма генетического программирования для задачи «Умный муравей-3».

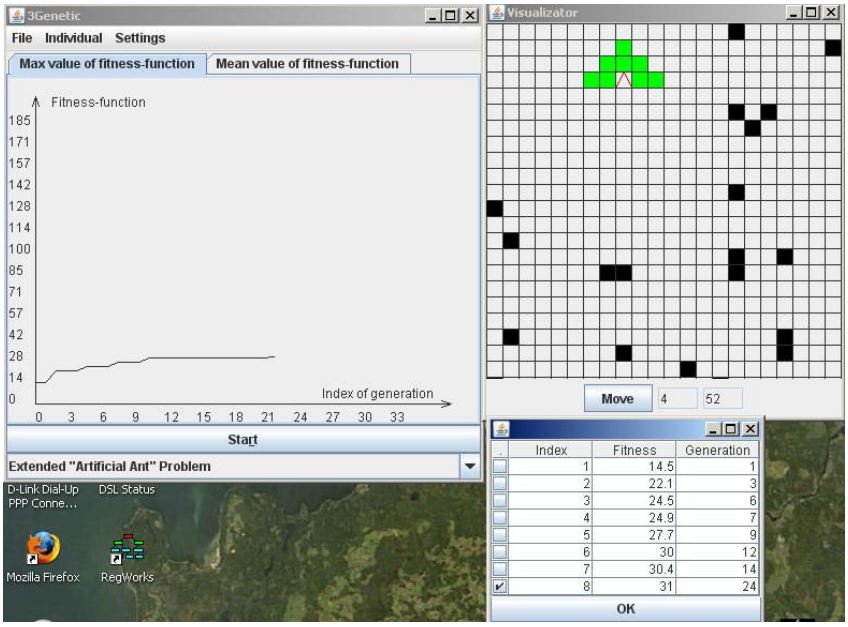

Рис. 2. Пример работы виртуальной лаборатории

На рис. 2 изображен график зависимости максимального значения функции приспособленности от номера поколения, а также визуализация работы лучшей особи.

На рис. 3 показано меню выбора алгоритма генетического программирования.

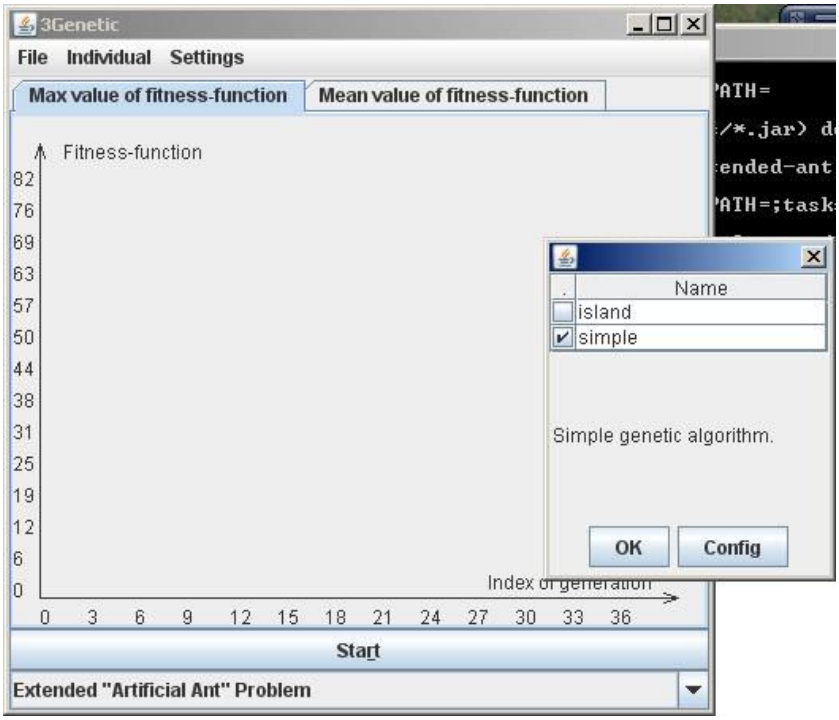

Рис. 3. Выбор генетического алгоритма

Выбор всех параметров, кроме решаемой задачи, осуществляется через `Меню -> Settings`.

Заключение

В описанную виртуальную лабораторию могут быть включены не только задачи, связанные с построением управляющих конечных автоматов, но и другие задачи по генетическим алгоритмам.

Предлагаемая система плагинов позволяет разделить решение задачи на части, что упрощает понимание сути генетического программирования, и ускоряет написание и тестирование решения.

Также система функций, позволяет сравнивать работу различных генетических алгоритмов по любым признакам.

В дальнейшем планируется добавить возможность работы с распределенными системами. Также планируется упростить систему добавления плагинов и интерфейса для описания задач.

Литература

1. Шалыто А.А. SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб. Наука, 1998. http://is.ifmo.ru/books/switch/1
2. Шалыто А. А. Технология автоматного программирования / Труды первой Всероссийской научной конференции «Методы и средства обработки информации» М.: МГУ. 2003. http://is.ifmo.ru/works/tech_aut_prog/
3. Jefferson D., Collins R., Cooper C., Dyer M., Flowers M., Korf R., Taylor C., Wang A. The Genesys System. 1992. www.cs.ucla.edu/~dyer/Papers/AlifeTracker/Alife91Jefferson.html
4. Angeline P. J., Pollack J. Evolutionary Module Acquisition // Proceedings of the Second Annual Conference on Evolutionary Programming. 2003. http://www.demo.cs.brandeis.edu/papers/ep93.pdf
5. Chambers L. D. Practical Handbook of Genetic Algorithms. V. 3. Algorithms to Improve the Convergence of a Genetic Algorithm with a Finite State Machine Genome. CRC Press, 1999.
6. Mitchell M. Introduction to Genetic Algorithms. MA: MIT, 1999.
7. Бедный Ю.Д., Шалыто А.А. Применение генетических алгоритмов для построения автоматов в задаче «Умный муравей» http://is.ifmo.ru/works/_ant.pdf
8. Яминов Б. Генетические алгоритмы http://rain.ifmo.ru/cat/view.php/theory/unsorted/genetic-2005