электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.832

Сравнение эффективности алгоритмов выхода из лабиринта на примере интеллектуальных агентов

***Аветисов А.А.**, студент*
кафедра «Информационные системы и телекоммуникации»
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана

***Богиня Н.В.**, студент*
кафедра «Информационные системы и телекоммуникации»
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана

Научный руководитель: Алфимцев А.Н., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
iu3@bmstu.ru

Введение

В наше время интеллектуальные алгоритмы нашли себе применение в задачах, которые сложно или невозможно решить без применения сложного поведения, обучаемости, автономности.

Например, к таким сложным задачам можно отнести:

- Управление роботами;
- Поиск информации в больших распределенных хранилищах;
- Логистика;
- Моделирование поведения людей;
- Предсказание погоды, катаклизмов, стихийных бедствий.

Кроме того, сегодня проводится много исследований на тему применения интеллектуальных алгоритмов. Примеры таких исследований:

- RoboCup [1] — международные соревнования среди роботов, основанные в 1993 году.
- Применение интеллектуальных алгоритмов в искусственном интеллекте роботов на турнирах Micromouse [2].
- Статья Пола Рейнера (Paul Reiner) «Роботы, лабиринты и архитектура поглощения» [3] о программировании виртуальных роботов на языке Java.

Перспективны также исследования применения интеллектуальных алгоритмов в областях космоса. Например, в статье NASA's OCA Mirroring System An application of multiagent systems in Mission Control [4] описана мультиагентная интеллектуальная система управления и слежения за Международной космической станцией - OCA Mirroring System (OCAMS).

Таким образом было бы логично предположить, что интеллектуальные алгоритмы будут находить все больше областей применения. Однако на практике, при выполнении различного рода задач, встречаются случаи, когда использование сложных интеллектуальных алгоритмов дает менее оптимальные результаты, нежели простые, тривиальные решения.

При разработке ПО разработчики постоянно сталкиваются с выбором сложности алгоритмов для выполнения поставленных целей [5]. То есть встает вопрос о целесообразности использования сложных, комплексных алгоритмов для решения различных задач.

Тема применения интеллектуальных алгоритмов для решения задачи прохождения лабиринта далеко не нова. Первые работы в этой области были опубликованы в середине 20 века К. Шенноном [6] и С. Клини [7].

1 Постановка задачи

В рамках курса «Мультиагентные интеллектуальные системы» кафедры «Информационные системы и телекоммуникации» МГТУ им. Н.Э. Баумана была поставлена следующая задача: провести эксперимент, в котором сравнить два алгоритма поиска выхода агента из лабиринта:

- на основе интеллектуального алгоритма выбора решения.
- на основе случайного выбора направления движения на развилке.

В результате сравнения нужно получить ответы на следующие вопросы:

A. Какое количество шагов в каждом случае понадобилось агенту для выхода из лабиринта?

B. На какой аппаратной платформе (указать характеристики Вашего персонального компьютера) проводился опыт и какое количество времени потребовалось в каждом случае на выход из лабиринта?

C. Какова верхняя оценка вычислительной сложности исследуемых алгоритмов? Приведите обоснование.

D. При всех ли размерах лабиринта найдется выход? Существует ли размерность лабиринта, при которой агент со случайным выбором будет плутать бесконечно?

В ходе выполнения описанной лабораторной работы студентами нашего потока использовались различные типы лабиринтов (рисунок 1), методы их построения, интеллектуальные алгоритмы поиска путей выхода из лабиринта (алгоритм одной руки, алгоритм Ли, волновой алгоритм, поиск в глубину и т.д.). Использовались также разные начальные условия (в рамках трактовки выданного задания), то есть в некоторых случаях, начальное положение агента определялось случайным образом, в других – положение агента как и выхода изначально были зафиксированы.

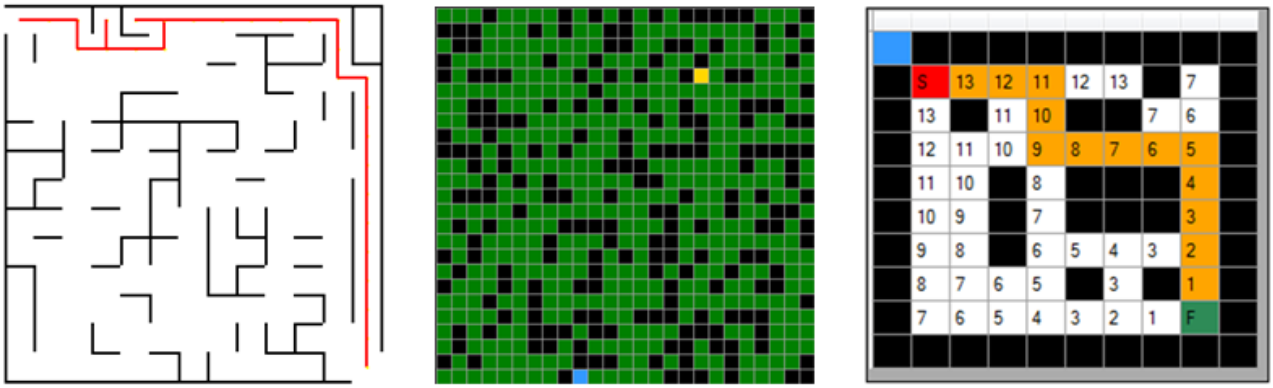


Рис. 1. Примеры лабиринтов, реализованных студентами нашей группы

Кроме того, алгоритмы были реализованы на языках программирования C#, C++, Java, Python, в различных средах разработки - Eclipse, Microsoft Visual Studio, gEdit, а испытания проводились на разных вычислительных платформах с разными операционными системами – от стационарных ПК с мощными процессорами Intel Core i7 до энергоэффективных платформ на базе Intel Atom. Таким образом, у многих студентов выводы о проделанной работе получились различные. Некоторые алгоритмы в сочетании со слабым аппаратным обеспечением вычислительного устройства, избыточностью кода, даже будучи интеллектуальными, действовали значительно медленнее случайных.

Несмотря на отличные друг от друга условия проведения эксперимента, было установлено следующее: при больших размерностях лабиринтов интеллектуальные алгоритмы имели явно выраженное преимущество, опережая случайных агентов в сотни и тысячи раз. Однако в случаях малых размерностей лабиринтов количество затрачиваемых шагов интеллектуальными агентами было соизмеримым, равным или даже превышало таковое для случайных агентов. Пронаблюдав эту интересную закономерность, мы

решили выяснить, при каком размере лабиринта случайный агент в принципе не сможет найти выход и насколько эффективней будет использовать интеллектуальные алгоритмы.

2 Описание программы

В качестве технологии реализации была выбрана агентная платформа JADE и язык программирования Java. Агентная платформа управляет жизненным циклом агентов, обеспечивает обмен сообщениями между агентами и поиск агентов.

Программа генерирует конфигурацию стенок лабиринта по алгоритму Hunt&Kill, что позволяет получить идеальный лабиринт («perfect maze») – односвязный, без петель и закрытых контуров и только одним маршрутом между двумя точками. Создаются четыре агента, три из которых – агенты-путники – ищут пути выхода из лабиринта, руководствуясь различными алгоритмами и полученной от четвертого агента-лабиринта информацией о конфигурации только той клетки, в которой они находятся.

Агент отправляет сообщение агенту-лабиринту (MazeAgent), в котором сообщает свои текущие координаты. MazeAgent отвечает сообщением о конфигурации стенок в этой клетке. На основании этой информации агент делает выбор следующей координаты и опять отправляет сообщение. Такой обмен происходит до достижения агентом выхода.

Агент-путник не знает изначально о координатах точки выхода, он даже не знает о размере лабиринта. Лишь попав в точку выхода непосредственно, агент выходит из лабиринта – эту ситуацию можно представить как проваливающийся пол в клетке выхода, который сработает только при нахождении на ней. Поведение всех агентов универсально для любых размеров карты и вычислительная сложность алгоритмов не зависит от размера лабиринта, потому что в каждый момент времени агенту приходится анализировать только одну клетку, а не весь массив.

Рассмотрим три алгоритма для поведения агентов-путников:

1. **RandomAgent** – случайный агент. Принятие решения о следующем шаге:
   a. Если клетка - не тупик, исключаем движение назад, в противном случае делаем шаг обратно;
   b. Смотрим конфигурацию стенок в текущей клетке и определяем возможные варианты движения;
   c. Выбираем направление случайным образом.
2. **IQAgent** – для этого агента в качестве алгоритма принятия решения было выбрано правило левой руки, которое можно успешно применять для используемого типа лабиринта. Принятие решения о следующем шаге:

a. Куда смотрит агент?(определить левое направление по отношению к нему);
b. Анализ конфигурации стенок:
    i. Если нет левой стенки, идем туда;
    ii. Если стенка слева есть, идем вперед;
    iii. Если есть стенка слева и впереди, идем вправо;
    iv. Если оказались в тупике, идем назад.
c. В каждом случае изменяется направление движения.

3. **IQAgent2** – агент с возможностью «телепорта» на клетку, координаты которой он предварительно запомнил. Путь, проходимый этим агентов всегда меньше, чем у агента IQAgent за счет исключения повторного прохода клеток. Принятие решения о следующем шаге:
    a. Если клетка – развилка, запомнить ее координаты и продолжать движение в любом направлении, выбранное направление запоминается;
    b. Если клетка – тупик, то в качестве следующего шага рассматривается следующая после последней развилки клетка, но в другом направлении, чем то, которое было выбрано ранее.

В интерфейсе программы можно задать размеры генерируемого лабиринта. По результатам прохождения агентами лабиринта на форму выводится информация о затраченном времени и количестве пройденных шагов.

На рисунке 2 представлены численные показатели прохождения лабиринта агентами и отображаются пути, пройденные агентами - путь IQAgent выделен красным цветом, путь агента Random – голубым, IQAgent2 – зеленым, кратчайший путь выделен желтым цветом.

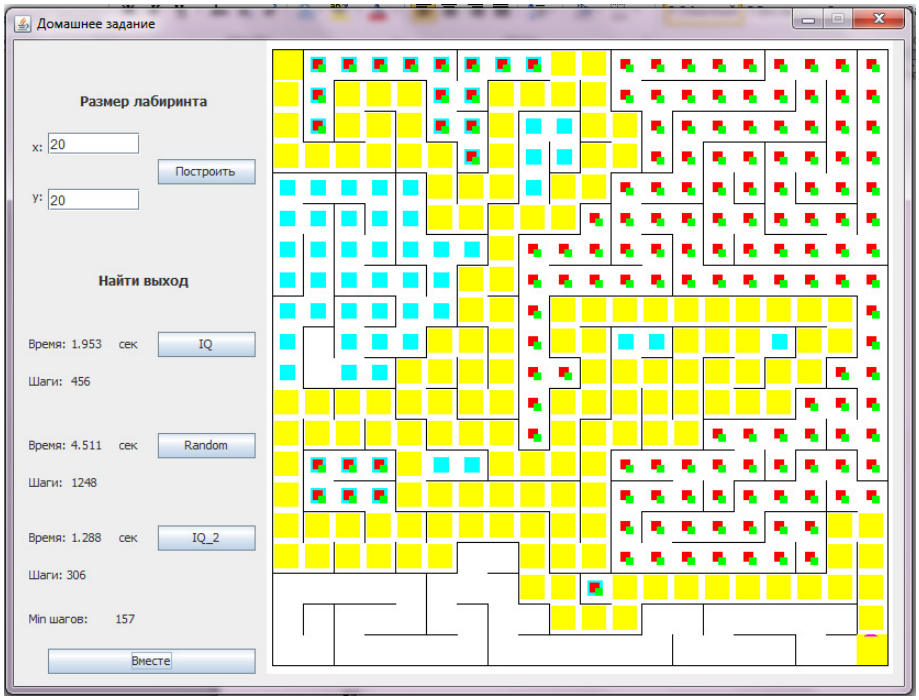


Рис.2. Интерфейс программы с графическим отображением путей, пройденных агентами

3 Постановка эксперимента

Была выдвинута следующая гипотеза, которую необходимо проверить с помощью эксперимента: с увеличением размера лабиринта увеличивается число шагов, необходимое каждому из агентов для нахождения выхода из лабиринта, причем для агента, запоминающего развилки, число шагов будет минимальным, а максимум будет наблюдаться для случая рандомного агента. Предполагается, что на очень больших размерах лабиринта (больше 150x150) рандомный агент может вообще не найти выход и плутать по лабиринту бесконечно.

Точки входа и выхода расположены в фиксированных местах – левом верхнем углу начальная точка и в правом нижнем – финальная точка (рисунок 2). Точки не ставятся случайно, чтобы исключить влияние этого фактора на результаты эксперимента. Фиксируя точки, мы добиваемся стабильной длины пути для каждого размера лабиринта.

Данные получены с помощью вышеописанной программы, причем один эксперимент заключается в прохождении по экземпляру (с уникальной конфигурацией стенок каждый следующий раз) лабиринта заданного размера по одному агенту каждого класса. Мы не стали пускать на экземпляре лабиринта по нескольку агентов, потому что для интеллектуальных агентов это не имеет смысла (по алгоритму, заложенному в них они могут идти только по одной траектории), а использование генератора псевдослучайных чисел java.util.Random не позволяет проводить различные эксперименты для рандомного агента.

Изначально было проведено по 10 экспериментов для десяти размеров лабиринта – от 10x10 до 100x100, но оказалось, что разброс среди каждой группы из десяти значений слишком высок, чтобы говорить об объективных средних значениях (рисунок 3). С учетом этого были проведены дополнительные эксперименты по рассмотренным случаям (их число увеличилось до 22 по каждой группе, причем отбрасывались минимальное и максимальное значения по критерию числа шагов рандомного агента, что дает нам по 20 значений в каждой группе), а также взяты дополнительные группы для промежуточных значений размеров лабиринта – от 15x15 до 95x95, а также рассмотрены случаи лабиринтов больших размеров – от 105x105 до 160x160.

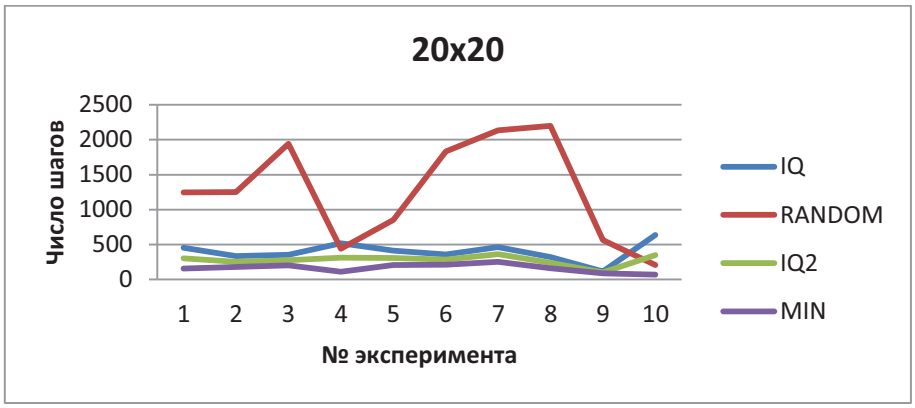


Рис. 3. Разброс числа шагов для 10 экспериментов при размере лабиринта 20x20

По каждому эксперименту записываются данные о затраченном времени и числе шагов для всех трех агентов, а также длина минимального пути от входа до выхода в шагах.

На больших размерах лабиринта рандомный агент тратит гораздо больше времени и шагов на прохождение – на один эксперимент уходит от 20 минут до нескольких часов при параметрах вычислительной платформы, представленной на рисунке 4, однако агент в конце концов находит выход.

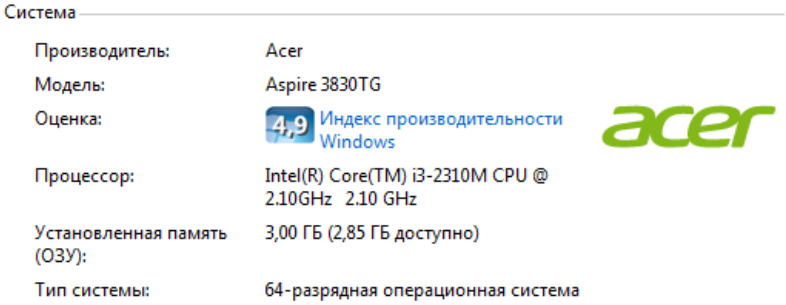


Рис. 4. Характеристики компьютера

4 Результаты эксперимента

Всего было поставлено по 22 эксперимента для 28 размеров лабиринта, то есть получено по 616 значений числа пройденных шагов для каждого агента. Данные были обработаны и проанализированы.

Оказалось, что при увеличении размера лабиринта резко возрастает число шагов и количество затраченного времени для случайного агента. Для интеллектуального агента эта зависимость носит не такой резкий характер – она плавно растет с увеличением размера лабиринта и среднего числа минимально возможных шагов. Но на минимальных размерах лабиринта RandomAgent в ряде случаев показывает результаты одинаковые или превосходящие результаты интеллектуального.

Среднее отклонение от среднего значения числа шагов в группе для одного размера лабиринта для интеллектуальных агентов не велика – составляет 26% от среднего значения для IQ и 22% для IQ2,а для случая рандомного агента – 61% от среднего.

Пытаясь определить параметры, которые влияют на то, пройдет ли рандомный агент длинным путем или коротким, мы пытались установить связь между числом шагов для интеллектуальных агентов и для рандомного на каждом экземпляре лабиринта. Предполагалось, что если интеллектуальные агенты затратят шагов больше среднего, то и от рандомного агента следует ожидать увеличения пути. Проанализировав следующие графики в совокупности (рисунок 5) и коэффициенты корреляции между зависимостями (таблица 1), можно утверждать, что шаги рандомного агента не зависят ни от числа шагов интеллектуальных агентов, ни от минимального числа шагов для выхода из лабиринта.

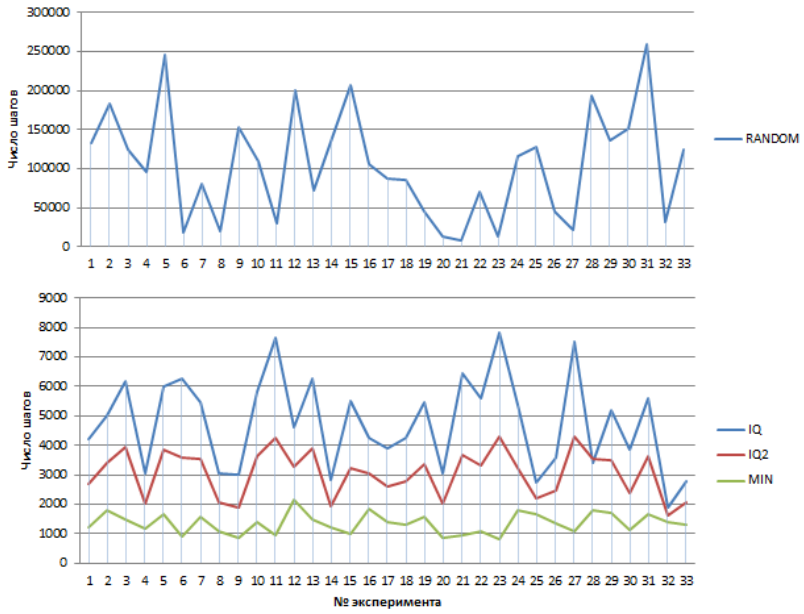


Рис. 5. Сравнение числа шагов для разных типов агентов для экспериментов при размере лабиринта 70x70

Таблица 1

Значение коэффициентов корреляции для зависимостей на рисунке 4

	Коэффициент корреляции		
iq	-0,148		
iq2	0,016	0,941	
min	0,537	-0,129	0,117
	random	iq	iq2

Другим параметром по данной проблеме могла бы выступать конфигурация лабиринта – насколько много в ней прямых линий (ведь рандомный агент идет только вперед), много ли перекрестков (именно здесь агент делает случайный выбор направления). При наличии влияния данного параметра предполагается наблюдать на простой конфигурации кратчайшего пути быстрое прохождение лабиринта, а на сложной – медленное. Для ответа на этот вопрос можно рассмотреть результаты работы программы на рисунке 6 – у лабиринта слева гораздо больше изгибов и развилок, однако там RandomAgent нашел выход быстрее, чем в случае, представленном справа (134328 шагов против 202122 шагов). Очевидно, что этот фактор не имеет решающего значения.

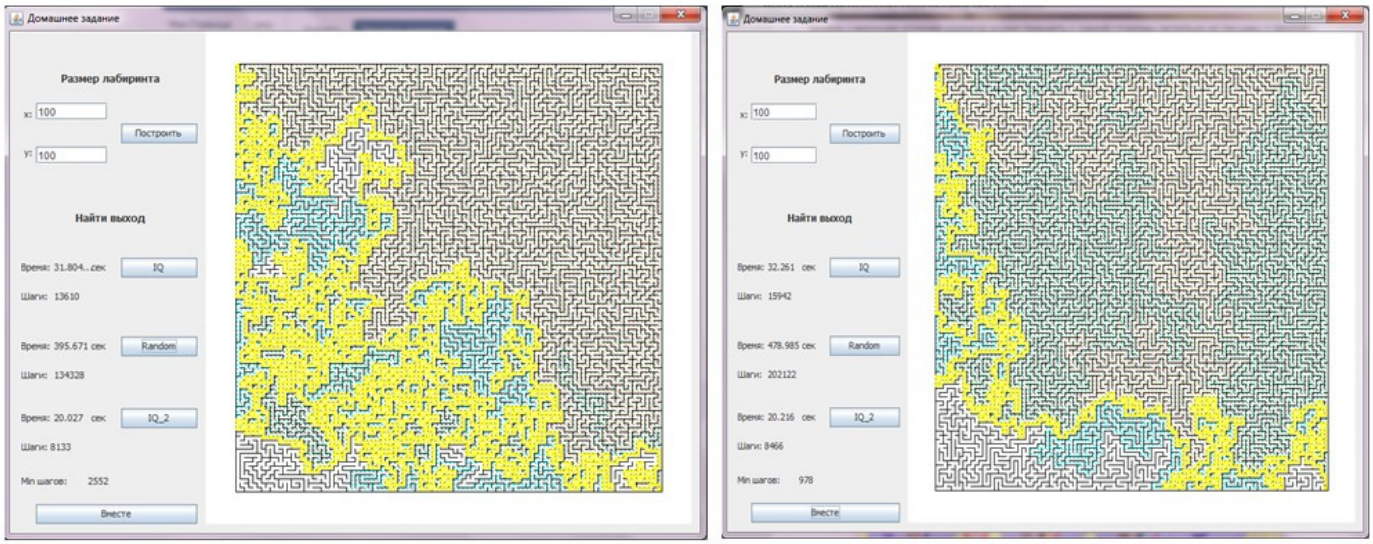


Рис. 6. Сравнение результатов прохождения агентами лабиринта 100x100 для различных конфигураций стенок

При увеличении выборки по каждой группе размеров лабиринта можно говорить о среднем значении числа шагов, необходимое агенту для нахождения выхода. Были рассмотрены 28 размеров лабиринта, таким образом у нас есть 28 средних значений, которые отмечены на рисунке 7.

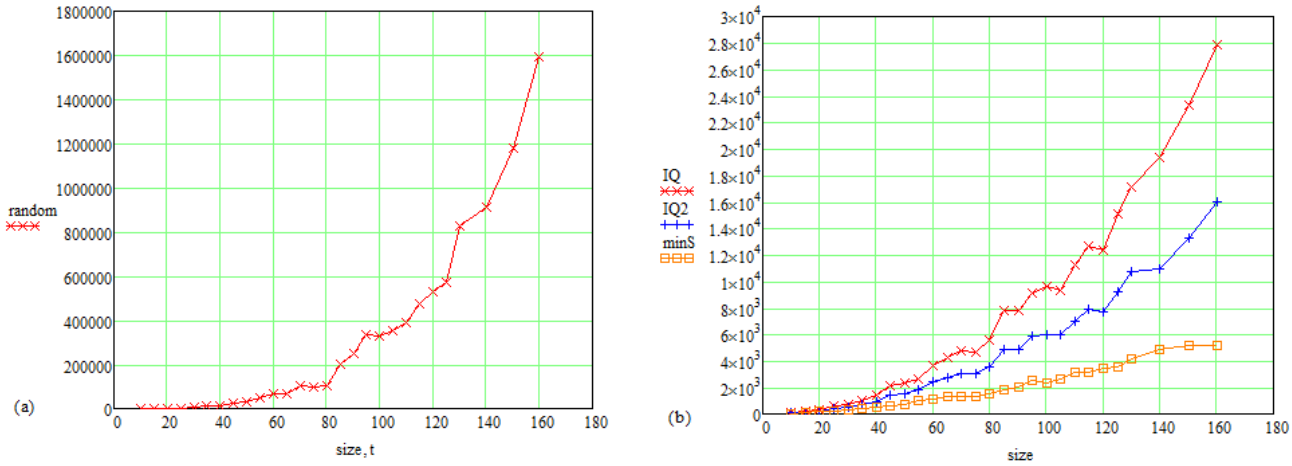


Рис.7. Результаты экспериментов: значения числа шагов для (a) - RandomAgent; (b) - IQAgent, IQ2Agent и кратчайшего пути

На рисунке 8 представлены графики с аппроксимирующими функциями, коэффициенты полинома и средняя ошибка аппроксимации. Для случаев интеллектуальных агентов и кратчайшего пути ошибка аппроксимации ниже 15%, что позволяет использовать полученные функции в качестве регрессии. Для агента RandomAgent удалось получить функцию с ошибкой аппроксимации только в 33%.

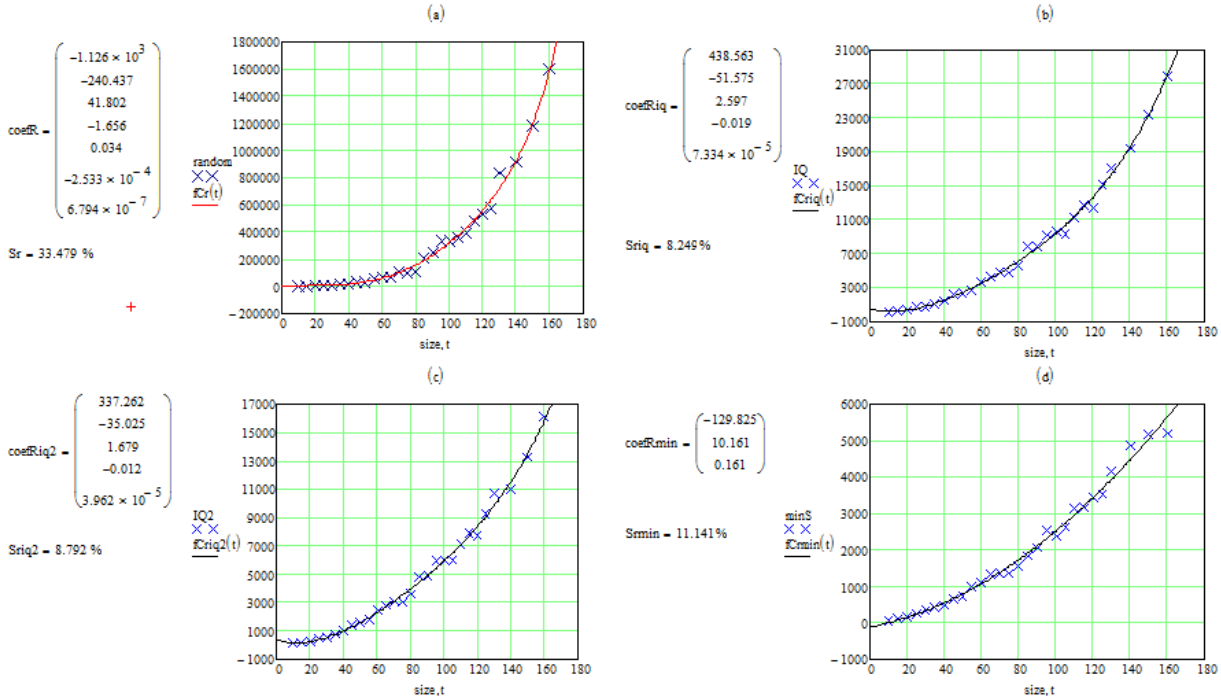


Рис. 8. Аппроксимация полиномами для функции агентов: (a) - RandomAgent; (b) - IQAgent;(c) - IQ2Agent; (d) - длина кратчайшего пути

На основании полученных уравнений регрессии мы можем сделать предсказание зависимости значений числа шагов для агентов и для длины кратчайшего пути. Графики

этих зависимостей приведены на рисунке 8. При размере лабиринта близком к 400x400 число шагов интеллектуальных агентов будет в 10^3 раз меньше, чем у рандомного агента и эта разница только увеличивается с расширением лабиринта. Длина кратчайшего пути увеличивается незначительно.

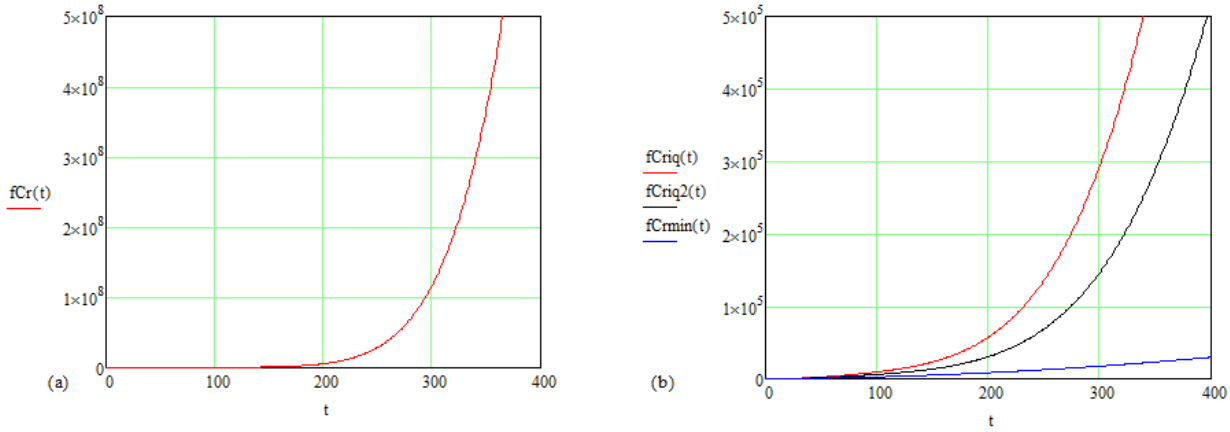


Рис. 9. Предсказание зависимости значений числа шагов от размера лабиринта для: (a) - RandomAgent;(b) - IQAgent, IQ2Agent и длины кратчайшего пути

5 Выводы

В ходе эксперимента, описанного в данной работе, были определены функции, которые описывают зависимость числа шагов агентов и длины кратчайшего пути от размера лабиринта. Показано, что увеличение размеров лабиринта приведет к серьезному увеличению числа шагов рандомного агента, однако мы не можем утверждать, что есть такой конечный размер лабиринта, при котором агент будет искать клетку выхода бесконечно долго, так как найденная аппроксимирующая функция не имеет вертикальной асимптоты.

Таким образом было установлено, что интеллектуальный алгоритм имеет преимущество перед алгоритмом простого перебора, причем в случаях тривиальных вычислительных задач, превосходство минимально или даже неочевидно. Однако при выполнении задачи, требующей большого количества вычислительной мощности необходимо применять интеллектуальные алгоритмы для достижения наилучших результатов.

Список литературы

1. Tandy M., Winkvist S., Young K. Competing in the RoboCup Rescue Robot League // Proc. Robotics for Risky Interventions and Surveillance of the Environment (RISE). Sheffield, 2010. Pp. 1-15.

2. Pandian J., Karthick R., Karthikeyan B. Maze Solving Robot Using Freeduino and LSRB Algorithm // International Journal of Modern Engineering Research (IJMER). 2013. №1. Pp. 92-100.
3. Рейнерс П. Роботы, лабиринты и архитектура поглощения. Режим доступа: http://www.ibm.com/developerworks/ru/library/j-robots/ (дата обращения 20.02.2014).
4. Sierhuis M. Clancey W. et. al. NASA's OCA Mirroring System: An application of multiagent systems in Mission Control // Proc. of Autonomous Agents and Multi Agent Conference (Industry Track). Budapest, 2009. Pp. 1-8.
5. Алфимцев А.Н. Разработка и исследование методов захвата, отслеживания и распознавания динамических жестов : дис. ... канд. техн. наук. М., 2008. 167 с.
6. Shannon C. E. Presentation of a maze-solving machine // Cybernetics Trans. of the 8th Conf. of the Josiah Macy Jr. Found / Editor: H. Forester. New York, 1952. Pp. 173-180.
7. Клини С. Представление событий в нервных сетях и конечных автоматах // Автоматы. М.: ИЛ, 1956. С. 15-67.