СПбГУТ)))
ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ТЕЛЕКОММУНИКАЦИИ
Июнь 2017 Том 5 № 2 ISSN 2307-1303 http://www.itt.sut.ru
TELECOM IT
June 2017 Vol. 5 Iss. 2 ISSN 2307-1303 http://www.itt.sut.ru

ВЕРОЯТНОСТНЫЙ МЕТОД ПОСТРОЕНИЯ МАШИНЫ СОСТОЯНИЙ ДЛЯ МОДУЛЯ ПРИНЯТИЯ РЕШЕНИЙ В РОБОТОТЕХНИЧЕСКИХ СИСТЕМАХ

Д. И. Михальченко[1], А. Г. Ивин[1*]

[1] СПИИРАН, Санкт-Петербург, 199178, Российская Федерация
* Адрес для переписки: arssivka@yandex.ru

Аннотация
На сегодняшний день популярность таких областей, как робототехника и разработка игр, значительно выросла, при этом достаточно велика степень взаимной интеграции этих областей. Так, в каждой из них требуется создание систем принятия решений. И наиболее часто используемым для этого в современных исследованиях и научных проектах инструментом являются машины состояний, хотя зачастую они не позволяют без больших трудозатрат создавать системы принятия решений достаточно эффективными. **Предмет исследования.** В данной статье предлагается модификация традиционного метода построения машин состояний на основе использования вероятностей, оценивающих события окружающей среды, в которой работает система принятия решений. **Основной результат.** Помимо этого, в статье предложен способ использования методов машинного обучения в сочетании с модифицированной машиной состояний. Также описана реализация фреймворка для принятия решений, встроенного в робототехническую платформу. **Практическая значимость.** Предложенная модификация позволяет достаточно просто создавать поведенческие реакции на ситуации, которые потребовали бы достаточно сложного решения при использовании традиционных машин состояний, а использование модифицированной машины состояний в сочетании с методами машинного обучение позволяет повысить эффективность разработанной системы принятия решений. Результаты подтверждены при помощи реализованной системы принятия решений для игрового симулятора.

Ключевые слова
робот, робототехника, игры, интеллект, решения, автоматы, вероятность, C++.

Информация о статье
УДК 004.89
Язык статьи – русский.
Поступила в редакцию 16.05.17, принята к печати 02.06.17.

Ссылка для цитирования: Михальченко Д. И., Ивин А. Г. Вероятностный метод построения машины состояний для модуля принятия решений в робототехнических системах // Информационные технологии и телекоммуникации. 2017. Том 5. № 2. С. 85–96.

PROBABILISTIC METHOD OF CONSTRUCTION OF THE STATE MACHINE FOR THE DECISION-MAKING MODULE IN ROBOTIC SYSTEMS

D. Mikhalchenko[1], A. Ivin[1*]

[1] SPIIRAS, St. Petersburg, 199178, Russian Federation
* Corresponding author: arssivka@yandex.ru

Abstract—At the moment popularity of robotics and game development is dramatically rising. One can see a sufficiently high degree of mutual integration of these two areas. For example, in almost every project related to these areas a decision-making system is required. The most popular way of creating decision-making systems is state machines. However, this way cannot always allow to create effective decision-making systems without high complexity. This paper introduces a method of creating state machines using probabilities for the description of situations in the external environment in which this system working. Also a way of combining machine-learning algorithms with modified state machine is presented in this paper. In addition, this paper includes presentation of framework for creating decision-making systems in robotic platform. Presented modification of state machines allows to create reactions of decision-making systems to the situations of external environment, which will require high labor costs while using traditional state machines, with ease. Combining this modified state machine with machine-learning algorithms allows to increase effectiveness of the created decision-making system. Results of the research are confirmed through practical realization of the proposed method in the test strategy for the gaming simulator.

Keywords—Robot, robotics, games, intellect, decisions, automates, probability, C++.

Article info
Article in Russian.
Received 16.05.17, accepted 02.06.17.

For citation: Mikhalchenko D., Ivin A.: Probabilistic Method of Construction of the State Machine for the Decision-Making Module in Robotic Systems // Telecom IT. 2017. Vol. 5. Iss. 2. pp. 85–96 (in Russian).

Введение

С распространением робототехники, а также непрекращающимся развитием области разработки игр (о чем, например, говорит возрастающая популярность чемпионата по игровому искусственному интеллекту Russian AI Cup[1]), возрастает степень интеграции технологий этих двух направлений. К примеру, системы принятия решений, разрабатываемые для компьютерных игр, могут успешно приме-

[1] Официальный свод правил Russian AI Cup 2016. URL: http://russianaicup.ru/p/rules (дата обращения 10.12.2016)

няться и в робототехнике. Наиболее распространенный выбор для создания системы принятия решений в робототехнических системах — конечные автоматы или машины состояний. Однако, как будет рассмотрено далее в работе, они не всегда достаточно эффективны. Целью работы является создание модифицированной машины состояний, позволяющей создавать более гибкие и эффективные системы принятия решений, не значительно усложняя их разработку и создание фреймворка, упрощающего реализацию систем принятия решения. На данный момент запланировано использование этого фреймворка в нескольких проектах, связанных с робототехническими системами и киберфизическим пространством [1]. Таким образом, выделяются следующие задачи, решаемые в работе: провести исследование в области создания систем принятия решений на основе машин состояний, модифицировать традиционный метод их создания, используя вероятностный подход и алгоритмы машинного обучения, провести тестирование модифицированной машины состояний и сравнительный анализ с традиционным ее вариантом, реализовать фреймворк для создания систем принятия решений.

1 Состояние исследований в изучаемой области

Несмотря на то что конечные автоматы – это довольно старая и хорошо изученная тема, по ней до сих пор существует достаточное количество актуальных исследований. Есть множество различных вариаций и типов алгоритмических конструкций, которые можно назвать конечным автоматом. Например, существуют так называемые взвешенные автоматы или преобразователи конечных состояний, которые, если говорить в терминах машины Тьюринга, имеют две ленты символов – входную и выходную [2]. Применение конечных автоматов и их вариаций достаточно широко. Благодаря наглядности и удобству разработки вероятностные конечные автоматы применяются во многих областях: распознавании сигналов [3], телемедицине [4], разработке компьютерных игр [5] и т. д. Однако при просмотре современных исследований в этой области можно заметить, что достаточно мало внимания уделяется новым идеям касательно разработки систем принятия решений и их улучшения. В большинстве случаев, как правило, используется обычный конечный автомат для описания поведения и редко что-то гораздо более сложное, как, например, популярные сегодня нейронные сети. Следует отметить применение конечных автоматов в робототехнических системах, к примеру, известная в кругах соревнований RoboCup команда из Германии BHuman использует стандартный конечный автомат для реализации поведений [6]. В работе [8] «объединение традиционного конечного автомата с современными подходами в виде нейронных сетей» используется только для распознавания изображений.

2 Модификация конечного автомата при помощи использования вероятностей

Обычно конечный автомат описывается в виде пятерки множеств $M = (V, Q, q0, F, f)$, где V — множество входных символов, Q — множество внутренних состояний, $q0$ — начальное состояние, f — функция переходов, которая определяет множество всех состояний, в которые из данного состояния возможен переход.

В предлагаемом варианте автомата, который в дальнейшем будет называться вероятностным, переходы осуществляются иным образом, а именно на основе вероятностных моделей. Обычные конечные автоматы мало пригодны для стохастических сред, в которых строго задать набор входных символов, определяющих систему принятия решений, представляется затруднительным. В данной работе описывается подход к созданию автоматов, для изложения которого следует ввести несколько вспомогательных терминов. «Мир ситуаций». Обобщая, можно сказать, что это понятие отвечает на вопрос «Что?». «Ситуация» показывает, что в окружающей среде с некоторой вероятностью произошло некое событие или набор событий. «Мир состояний». Это понятие отвечает на вопрос «Как?». Состояния описывают то, каким образом система принятия решений должна себя вести. Каждое состояние представляет собой определенный набор действий, которые должна совершить система. «Связь», «переход» или «отношение». По аналогии с переходами в обычных конечных автоматах, связи или отношения описывают, какие «что?» связаны с какими «как?». Такой подход предполагается более уместным для стохастических сред, особенно для сред, в которых необходимо принимать решения в настоящем времени. Он позволяет более гибко описывать окружающую среду, а также избежать необходимости ее строгого описания. Однако в таком автомате возникает определенная сложность при описании переходов. Если представить себе, что в описываемой системе принятия решений возникла ситуация, когда два перехода равновероятны, то в таком случае каким-то образом нужно выбрать один из возможных вариантов. Сложно назвать такой выбор рациональным. Если говорить об игровых ситуациях, то можно себе представить, что в какой-то момент возникли две равновероятные ситуации «Союзная база под атакой» и «Присутствие бонуса». В таком случае случайный выбор может привести к реагированию на ситуацию «Присутствие бонуса», в то время как уничтожение союзной базы приводит к проигрышу в принципе. Решение, предлагаемое в данной работе, – это введение и использование такого понятия, как факторы. Фактор – это по своей сути число, на которое умножается вычисленная вероятность ситуации. Таким образом, значение вероятности можно понижать или повышать, ранжируя ситуации по важности. Это позволит избежать ситуаций неопределённого выбора, а также сделать стратегию более эффективной. У введения понятия факторов есть еще одно достоинство: к такому автомату можно применять методы машинного обучения. Легко заметить, что верно проранжированные при помощи факторов переходы способны определить то, насколько эффективной будет стратегия. Более того, две абсолютно одинаковых стратегии, но с разным набором факторов, могут при практическом применении показывать совершенно разные поведения. Поскольку ранжирование факторов вручную может оказаться весьма трудозатратным и малоэффективным процессом, машинное обучение конечного автомата значениям факторов представляется наиболее перспективным средством повышения эффективности его стратегии. Как уже было сказано, в данном подходе все события во внешнем для системы принятия решений мире происходят с некой вероятностью. Произведение вероятностей определенных событий представляет собой переход внутри конечного автомата. События могут быть использованы повторно

для создания новых переходов. По теореме умножения вероятностей можно описать перечисленное в виде формул (1), (2).

$$p_{event\_i} = f(args), \quad (1)$$

$$p_{transition} = \prod_0^n p_{event\_i}, \quad (2)$$

где p_{event_i} – вероятность возникновения i-го события, $f(args)$ – функция, описывающая вероятность возникновения события, $p_{transition}$ – вероятность возникновения перехода. То есть, переход – ситуация, состоящая из набора событий, определяется как произведение этих событий.

3 Реализация системы принятия решений с использованием вероятностного автомата

3.1 Описание среды

В качестве тестовой площадки для реализации вероятностного автомата использовался симулятор, поставляемый организаторами соревнования Russian AI Cup 2016[2]. Использование игрового симулятора для тестирования способа создания систем принятия решений обусловлено тем, что в принципе системе принятия решений неважно, работает ли она в виртуальной или реальной среде. К тому же зачастую при разработках робототехнических систем сначала эти системы тестируют в симуляторах, как например в [8]. Также реализация именно игровой стратегии не сильно отличается от реализации стратегии для робототехнической системы, поскольку что игровая стратегия, что робототехническая система – это автономная система, которой необходимо совершать определенны действия в определенной среде. Им приходиться решать в большинстве случаев одни и те же проблемы: решения о том, в какую позицию переместиться, какое совершить действие в этой позиции и тому подобное. Таким образом, тестирование системы принятия решения в игровом симуляторе можно считать актуальным.

Чемпионат пятого Russian AI Cup называется CodeWizard. Всем участникам соревнования необходимо запрограммировать искусственный интеллект для управления «волшебником». Правила соревнования базируются на популярном в мире компьютерных игр жанре «MOBA» — играх, сочетающих в себе элементы стратегии в реальном времени и ролевой игры, в которых две команды игроков сражаются друг с другом на ограниченной карте. Подробное описание правил игры и целей игрока можно найти на сайте[3] [1].

3.2 Описание стратегии

Была реализована стратегия, которая должна действовать в описанной игровой среде с использованием модифицированного автомата. В этой стратегии

[2] Официальный свод правил Russian AI Cup 2016. URL: http://russianaicup.ru/p/rules (дата обращения 10.12.2016)
[3] См. выше

при помощи вероятностей оцениваются следующие ситуации: вероятность существования бонуса, которая описывается квадратичной функцией $p = \left(\frac{rest - \frac{per}{2}}{2}\right)^2$, зависящей от периода генерации бонуса (*per*) в тиках и того, сколько тиков прошло с момента прошлой генерации (*rest*). Если бонус присутствует на позиции, то устанавливается максимальная вероятность данной ситуации. Такой расчет необходим для того, чтобы была возможность начать движение к бонусу еще до его непосредственного появления, чтобы успеть его подобрать. Вероятность того, что дружественные строения находятся под атакой; вероятность того, что дружественные юниты производят атаку на вражеское строение. Для оценки вероятности того, что строение находится под атакой, независимо от того, какой команде оно принадлежит оценивается следующим образом: юниты кластеризуются в группы, происходит оценка расстояний от групп юнитов до строений. Чем ближе группа к области атаки строения, тем выше вероятность конкретной ситуации; вероятность необходимости отступать: вероятность необходимости отступать зависит от количества. Все факторы оставались равными 1.

4 Использование алгоритмов машинного обучения в совмещении с модифицированным конечным автоматом

4.1 Описание ситуации для обучения и выбор алгоритма

Для обучения факторов вероятностного автомата использовался генетический алгоритм. Генетические алгоритмы решают задачу оптимизации, используя механизмы, аналогичные естественному отбору в природе: отбор, размножение и мутации. Они достаточно просты в реализации и идеально подходят для решения типовой задачи обучения факторов переходов вероятностного автомата. Несмотря на то что первые работы, посвященные разработке и исследованию генетических алгоритмов, появились довольно давно, этот подход и сегодня не утратил своей актуальности [9]. В обучении будет использоваться автомат, реализованный для игровой стратегии Russian AI Cup. Реализовать обучение возможно благодаря использованию симулятора, предоставляемого командой Russian AI CUP для реализации стратегий.

4.2 Реализация алгоритма обучения

Алгоритм обучения имеет следующий вид: 1. Загрузить необходимые значение факторов. 2. Запустить симулятор. 3. Запустить стратегию с текущим набором факторов. 4. После завершения игры подсчитать и сохранить количество очков. 5. Повторять пункты 1–4 для всех особей поколения. 6. По завершению обработки всех особей поколения, сохранить результат и факторы лучшей особи текущего поколения, произвести переход к новому поколению и вернуться к пункту 1. 7. Когда обработка всех поколений завершена, имеется лог-файл с результатами работы: списком параметров лучших особей из каждого поколения и их результатов. Был создан класс, реализующий работу самого генетического алгоритма.

При создании класса происходит инициализация первого поколения. Инициализация происходит при помощи случайных чисел, а значение «фитнесс-функции» для всех особей выставляется равным 0. Для размножения используются две случайные особи, прошедшие отбор по наибольшему значению «финтесс-функции». От каждой особи берется по половине факторов от общего их количества. Отбор проходит только половина всех особей. Мутации происходят для определенного заранее заданного количества особей. Выбирается случайная особь, а также случайное количество факторов, которые подвергнутся мутации. Далее выбранные факторы принимают случайное значение.

4.3 Тестирование стратегий

Для наглядности результатов обучения и анализа эффективности полученной в результате стратегии было произведено по 10 тестовых игр для стратегии с факторами, равными 1, и для стратегии с факторами, полученными в результате обучения. Также, для сравнения была реализована стратегия, использующая традиционный конечный автомат, но описание ее реализации выходит за рамки данной статьи.

В результате обучения были получены следующие факторы: подбор верхнего бонуса – 0,8; подбор нижнего бонуса – 0,5; отступление – 0,9; атаковать первую верхнюю башню – 0,3; атаковать вторую верхнюю башню – 0,0; атаковать первую правую башню – 0,1; атаковать вторую правую башню – 0,3; атаковать первую центральную башню – 0,9; атаковать вторую центральную башню – 0,3; атаковать вражескую базу – 1,0. Для упрощения восприятия значений факторов присутствуют нормированные значения в диапазоне от 0 до 1.

По полученным факторам при знании правил игры видно, что стратегия направлена на получение максимально возможного количества очков.

В таблице 1 представлены статистические данные по результатам десяти игр для необученной стратегии, использующей модифицированный автомат. В таблице 2 аналогичным образом представлены результаты игр обученного автомата. Место – место, занятое стратегией среди остальных по количеству очков. Выигрыш – вражеская база была уничтожена, проигрыш – была уничтожена союзная база. Игра останавливается по прошествии 20 000 тиков, если не было выигрыша или проигрыша, поэтому в таблице есть ситуации, в которых нет выигрыша или проигрыша.

Таблица 1.

Результаты игр необученного автомата

Очки	**2043**	**3030**	**2679**	**2200**	**3021**	**2591**	**2339**	**2495**	**2443**	**2515**
Место	6	1	1	6	1	3	4	4	6	1
Выигрыш	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Да
Проигрыш	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Да	Нет

Таблица 2.

Результаты игр обученного автомата

Очки	**3200**	**3016**	**3365**	**3331**	**2365**	**3153**	**2535**	**3080**	**2117**	**3277**
Место	1	1	1	1	6	1	1	3	1	1
Выигрыш	Нет	Нет	Да	Нет	Нет	Нет	Нет	Да	Да	Нет
Проигрыш	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет

По этим двум таблицам можно наблюдать увеличение эффективности стратегии при использовании обученной версии. Значительно возросло количество первых мест.

В таблице 3 продемонстрировано сравнение усредненных результатов после 10 игр обученного и необученного автомата.

Таблица 3.

Сравнительные результаты обученного и необученного автомата

Стратегия	**Среднее кол-во очков**	**Усреднённое место**	**Выигранные игры, %**	**Проигранные игры, %**
Необученная	2 535,6	3,3	10	10
Обученная	2 943,9	1,7	30	0

По усредненным значениям наглядно видно, насколько эффективнее обученная стратегия по сравнению с необученной. Прирост очков ~400 мог бы положительно повлиять на место, занимаемое стратегией среди стратегий других игроков.

Для большей наглядности в таблице 4 представлены усредненные результаты 10 игр других удачных особей из других поколений, в том числе финального – 100-го.

Таблица 4.

Усредненные результаты различных поколений

Номер поколения	**Среднее кол-во очков**	**Усреднённое место**	**Выигранные игры, %**	**Проигранные игры, %**
81	3001,4	1,6	30	10
100	3022,5	1,5	20	0

Результаты лучших особей поколения 81 и поколения 100 практически идентичны по своим значениям, хоть и за счет отсутствия проигрышей стратегию, ос-

нованную на особи 100-го поколения, можно назвать более эффективной. По полученным результатам видно, что обученный автомат заметно эффективнее необученного. Все протестированные обученные стратегии по количеству очков в среднем ближе к первому месту среди стратегий, задействованных в игре. Кроме того, обучение значительно повышает вероятность уничтожения вражеской базы, что является одной из целей в игре.

Тестирование стратегии на традиционном автомате показало средний результат полученных очков: 2 383,7, равный процент выигрышей и проигрышей: 10 %. Сравнивая эти данные с результатами таблиц 1–4, несложно заметить, что результаты этой стратегии значительно ниже результатов стратегии использующей модифицированный конечный автомат.

5 Реализация фреймворка принятия решений для внедрения в робототехническую платформу

В фреймворке предполагается использование двух видов машин состояний: детерминированные автоматы и модифицированные автоматы. Таким образом, фреймворк должен предоставлять функционал для реализации любой из этих машин. У обоих видов машин есть общие элементы или схожие по своему назначению. На рисунке изображён прототип фреймворка и наглядно показано использование общих частей.

Основные концепции, используемые в фреймворке: хранилище данных – объект, который получает все необходимые данные из среды, при необходимости обрабатывает; генератор события – имея доступ к хранилищу данных, генерирует либо соответствующую данным ситуацию, произошедшую в среде, либо набор вероятностей возможно произошедших ситуаций; алгоритм связи – определяет связь между набором условий и ситуацией; действие – некое базовое действие, которое способна совершать система, реагируя на условия внешней среды; ситуация – абстракция, соответствующая набору определенных условий либо вероятностей, описывающая среду на данный момент; условие – абстракция, определяющая, что некое поле данных в хранилище данных имеет определенное значение.

Для реализации конечного варианта фреймворка был использован язык C++, поскольку предполагается его интеграция в робототехническую платформу, описываемую в [10, 11]. Данный фреймворк разработан с учетом специфики использования в робототехнических системах, а также является пригодным для использования в игровых системах, грубо говоря, в тех средах, где все исполнение построено на вычислении действий в определенный «фрейм». Реализованный фреймворк активно использует механизмы 11-го стандарта языка C++ [12] в результате чего не зависит от сторонних библиотек.

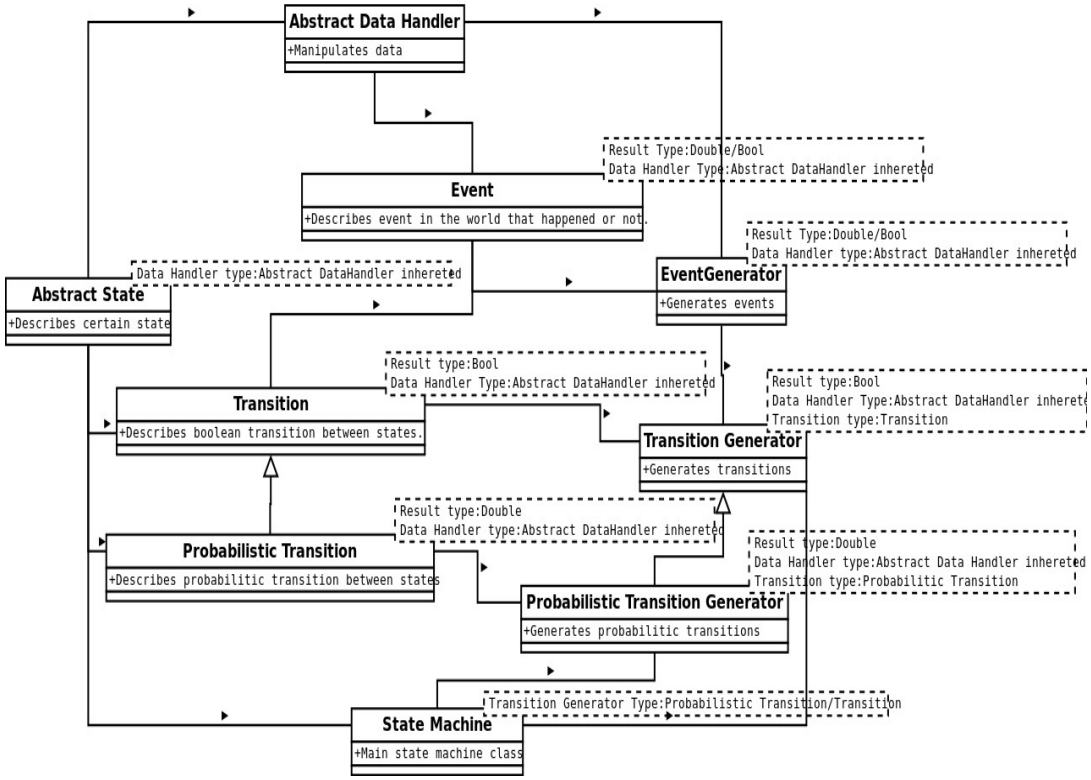

Рисунок. Прототип фреймворка

Заключение

В работе были предложены следующие модификации: описание переходов при помощи произведения вероятностей событий, которые должны вызвать переходы; введение так называемых факторов для ранжирования вероятностей и возможности использования алгоритмов машинного обучения для увеличения эффективности системы принятия решений, реализованной при помощи модифицированного конечного автомата. Эффективность использования такого подхода была продемонстрирована при помощи результатов реализованной стратегии для Russian AI Cup. Использование инструментов машинного обучения было успешным и позволило повысить эффективность реализованной стратегии.

Литература

1. Левоневский Д. К., Ватаманюк И. В., Савельев А. И., Многомодальная информационно-навигационная облачная система МИНОС для корпоративного киберфизического интеллектуального пространства // Программная инженерия. 2017. № 3. С. 120–128.

2. Mohri M. Weighted automata algorithms //Handbook of weighted automata. – Springer Berlin Heidelberg. 2009. С. 213–254.

3. Mukherjee K., Ray A. State splitting and merging in probabilistic finite state automata for signal representation and analysis // Signal processing. 2014. Vol. 104. pp. 105–119.

4. Sim I. et al. Clinical decision support systems for the practice of evidence-based medicine //Journal of the American Medical Informatics Association. 2001. Vol. 8. No. 6. pp. 527–534.

5. Bjork S., Holopainen J. Patterns in game design (game development series). 2004.

6. BHuman publications. URL:_https://www.bhuman.de/publications-en.html (дата обращения: 02.09.2016)

7. Савченко О. В., Куреннов Д. В. Система управления роботом на основе конечного автомата и нейронной сети // Вестник УГАТУ. 2014. № 5. URL: https://cyberleninka.ru/article/n/sistema-upravleniya-robotom-na-osnove-konechnogo-avtomata-i-neyronnoy-seti (дата обращения: 23.04.2017).

8. Jeon S., Lee J. Framework and Modeling of a Multi-robot Simulator for Hospital Logistics // Advances in Intelligent Systems and Computing. 2017. Vol. 447. pp. 213–219.

9. Nourmohammadzadeh A., Hartmann S. The Fuel-Efficient Platooning of Heavy Duty Vehicles by Mathematical Programming and Genetic Algorithm // Lecture Notes in Computer Science. 2016. Vol. 10071. pp. 46–57.

10. Ivin A., Mikhalchenko D. Software Platform for Development of Multimodular Robotic Systems with Asynchronous Multithreaded Control // 20th Conference of Fruct Association. 2017. pp. 1–7.

11. Denisov A., Budkov V., Mikhalchenko D., Designing Simulation Model of Humanoid Robot to Study Servo Control System // Lecture Notes in Computer Science. 2016. Vol. 9812. pp. 69–78.

12. Meyers S. Effective Modern C++. O'Reilly Media Inc. 2014. 117 p.

References

1. Levonevskiy D., Vatamaniuk I., Saveliev A., MINOS Multimodal Information and Navigation Cloud System for the Corporate Cyber-Physical Smart Space // Programmnaya inzheneriya. 2017. No. 3. pp. 120–128.

2. Mohri M. Weighted automata algorithms //Handbook of weighted automata. – Springer Berlin Heidelberg. 2009. C. 213–254.

3. Mukherjee K., Ray A. State splitting and merging in probabilistic finite state automata for signal representation and analysis // Signal processing. 2014. Vol. 104. pp. 105–119.

4. Sim I. et al. Clinical decision support systems for the practice of evidence-based medicine //Journal of the American Medical Informatics Association. 2001. Vol. 8. No. 6. pp. 527–534.

5. Bjork S., Holopainen J. Patterns in game design (game development series). 2004.

6. BHuman publications. URL:_https://www.bhuman.de/publications-en.html (дата обращения: 02.09.2016)

7. Savchenko O., Kurennov D. Robot Control System on the basis of Finite Automaton and the Neural Network // Vestnik UGATU. 2014. No. 5. URL: https://cyberleninka.ru/article/n/sistema-upravleniya-robotom-na-osnove-konechnogo-avtomata-i-neyronnoy-seti (Access Date: 23.04.2017).

8. Jeon S., Lee J. Framework and Modeling of a Multi-robot Simulator for Hospital Logistics // Advances in Intelligent Systems and Computing. 2017. Vol. 447. pp. 213–219.

9. Nourmohammadzadeh A., Hartmann S. The Fuel-Efficient Platooning of Heavy Duty Vehicles by Mathematical Programming and Genetic Algorithm // Lecture Notes in Computer Science. 2016. Vol. 10071. pp. 46–57.

10.Ivin A., Mikhalchenko D. Software Platform for Development of Multimodular Robotic Systems with Asynchronous Multithreaded Control // 20th Conference of Fruct Association. 2017. pp. 1–7.

11.Denisov A., Budkov V., Mikhalchenko D., Designing Simulation Model of Humanoid Robot to Study Servo Control System // Lecture Notes in Computer Science. 2016. Vol. 9812. pp. 69–78.

12.Meyers S. Effective Modern C++. O'Reilly Media Inc. 2014. 117 p.

Михальченко Даниил Игоревич – младший научный сотрудник, СПИИРАН, Санкт-Петербург, 199178, Российская Федерация, ocnechlahim@mail.ru

Ивин Арсений Григорьевич – младший научный сотрудник, СПИИРАН, Санкт-Петербург, 199178, Российская Федерация, arssivka@yandex.ru

Mikhalchenko Daniil – Research Assistant, SPIIRAS, St. Petersburg, 199178, Russian Federation, ocnechlahim@mail.ru

Ivin Arseniy – Research Assistant, SPIIRAS, St. Petersburg, 199178, Russian Federation, arssivka@yandex.ru