УДК 004.852

А.А. Бессонов

ХНЭУ им. С. Кузнеца, г. Харьков, Украина, oleksandr.bezsonov@hneu.net

СОСТАВЛЕНИЕ РАСПИСАНИЯ СПОРТИВНЫХ ТРЕНИРОВОК С ПОМОЩЬЮ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

Статья посвящена разработке программного обеспечения на основе генетических алгоритмов (ГА), которое осуществляет планирование теннисных тренировок с участием игроков различного пола, возраста и опыта. Выбор ГА обусловлен тем, что данные алгоритмы хорошо зарекомендовали себя при решении подобных задач планирования, являются достаточно исследованными, обладают высоким быстродействием и обеспечивают получение решений, близких к оптимальным. Приводятся результаты работы разработанного приложения, подтверждающие эффективность предложенных алгоритмов составления расписания.

РАСПИСАНИЕ ТРЕНИРОВОК, ГЕНЕТИЧЕСКИЙ АЛГОРИТМ, ФИТНЕС-ФУНКЦИЯ, СКРЕЩИВАНИЕ, МУТАЦИЯ

Безсонов О.О. Складання розкладу спортивних тренувань за допомогою генетичних алгоритмів. Стаття присвячена розробці програмного забезпечення на основі генетичних алгоритмів (ГА), яке здійснює планування тенісних тренувань за участі гравців різної статі, віку і досвіду. Вибір ГА обумовлений тим, що дані алгоритми добре зарекомендували себе при вирішенні подібних завдань планування, є досить дослідженими, мають високу швидкодію і забезпечують отримання рішень, близьких до оптимальних. Наводяться результати роботи розробленого додатка, що підтверджують ефективність запропонованих алгоритмів складання розкладу.

РОЗКЛАД ТРЕНУВАНЬ, ГЕНЕТИЧНИЙ АЛГОРИТМ, ФІТНЕС-ФУНКЦІЯ, СХРЕЩУВАННЯ, МУТАЦІЯ

Bezsonov O.O. Sports training scheduling using genetic algorithms. The article is devoted to the development of software application based on genetic algorithms (GA), which performs the scheduling of tennis trainings with the participation of players of different sex, age and experience. The choice of GA is due to the fact that these algorithms have proven themselves in solving such scheduling problems. GA are sufficiently researched, have high speed and provide solutions that are close to optimal. The presented simulation results confirm the effectiveness of the proposed algorithms for scheduling.

TRAINING SCHEDULE, GENETIC ALGORITHM, FITNESS FUNCTION, CROSSING, MUTATION

Введение

Многие практические и теоретические проблемы оптимизации характеризуются многомерностью пространства поиска. Эти проблемы включают в себя NP-полные задачи комбинаторной оптимизации, идентификации сложных структур или многомерной оптимизации функций и т.д. Наиболее часто такие проблемы возникают в области планирования.

Планирование – одна из многих широко изученных и исследованных практических задач в комбинаторной оптимизации. Планирование состоит из разработки расписания некоторых событий в пределах заданного временного интервала и предоставления конечного набора ресурсов таким образом, чтобы данные события не конфликтовали друг с другом и не нарушали при этом заданных ограничений. Проблемы с составлением расписания возникают во многих областях, таких как планирование заданий на производстве, планирование дежурств в больнице, планирование процессов в компьютерной системе, планирование выпускных экзаменов в школе или университете. В большинстве случаев задачи планирования не имеют известного алгоритма решения полиномиальной сложности, способного найти решение, которое оптимально сочетает данные ресурсы без нарушения ограничений. Пространство поиска, состоящее из всех комбинаций событий, времени и местоположений, может быть слишком большим для тщательного поиска в разумные сроки. Во многих реальных приложениях планирования эта работа выполняется вручную экспертами, использующими многолетний опыт для нахождения близких к оптимальным или наиболее благоприятных решений. Этот процесс занимает значительное время и конечный результат обычно далек от желаемого. Чтобы устранить неэффективную практику ручного планирования, исследователи пытались автоматизировать данную задачу с использованием вычислительной техники, которая теоретически позволяет сгенерировать расписание за значительно меньшее время, чем человек. Однако, как и во многих других случаях, связанных с эвристикой, кодирование человеческого знания в компьютеризированной форме оказалось довольно сложной задачей. Тем не менее, с целью автоматизации сложных задач планирования были исследованы различные методы, результаты которых показали, что применение традиционных методов, таких как

градиентные, динамическое программирование, симплекс-метод для решения этой задачи часто не дает желаемого результата, так как вычислительные затраты растут экспоненциально вместе с размерностью задачи. В связи с этим довольно часто для решения практических задач применяются эвристические методы с гораздо более низкими вычислительными затратами, несмотря на то, что они могут и не обеспечивать достижения глобального оптимального решения.

Около трех десятилетий назад в литературе начали обсуждать заимствованные у природы эвристические методы, которые оказались более гибкими и эффективными. К таким методам оптимизации относят метод имитации отжига, который проводит аналогию между отжигом материала до его самого низкого энергетического состояния и решением задачи оптимизации, эволюционные алгоритмы (ЭА), в основном заимствованные из биологической эволюции. Такие современные подходы, как табу поиск, муравьиные алгоритмы, метод роя частиц также упоминаются в контексте заимствованных у природы методов оптимизации. В последнее время получает все большее значение в области эвристической оптимизации теория агентов [1]. В данной работе основное внимание уделяется такому разделу эволюционных вычислений, как генетические алгоритмы (ГА), с помощью которых осуществляется планирование теннисных тренировок между игроками различного пола, возраста и опыта. Выбор ГА обусловлен тем, что данные алгоритмы хорошо зарекомендовали себя при решении подобных задач планирования, являются достаточно исследованными, обладают высоким быстродействием и обеспечивают получение решений, близких к оптимальным.

1. Генетические алгоритмы

ГА являются методом машинного обучения, использующим механизмы отбора в природе [2], осуществляющие случайный и параллельный поиск решений, которые оптимизируют заранее определенную фитнес функцию [3]. ГА были впервые предложены в Мичиганском университете американским исследователем Дж. Холландом [4-5] для решения задач оптимизации в качестве достаточно эффективного механизма комбинаторного перебора вариантов решения. В отличие от многих других работ, целью Холланда было не только решение конкретных задач, но и исследование явления адаптации в биологических системах и применение его в вычислительных системах. При этом потенциальное решение (особь) представляется хромосомой – двоичным кодом. Популяция содержит множество особей. В процессе эволюции используются три основных генетических оператора: репродукция, кроссинговер и мутация. Голдберг (ученик Холланда) успешно развил ГА и расширил области их применения. Его монография [6], в которой систематически изложены основные результаты и области практического применения ГА, является в настоящее время наиболее известной и цитируемой.

Основываясь на идеях Холланда, была принята концепция стандартного генетического алгоритма (SGA), на которую сильно повлиял его биологический архетип [7]. В связи со значительным ростом вычислительной мощности с 1975 года, потенциал ГА увеличивается все больше и больше. В связи с этим популярность GA-концепций неуклонно растет, и многие ученые по всему миру стали решать различные проблемы науки и техники с помощью ГА. Однако вскоре стало очевидным, что для большинства практических задач двоичное кодирование, первоначально предложенное Холландом, вовсе не является достаточным. В связи с этим были разработаны различные варианты кодировки информации, а также новые операторы скрещивания и мутации, применение которых весьма разнообразило поведение проектируемых на основе ГА приложений. Обзор различных кодировок и операторов, разработанных для различных приложений, может быть найден в [8]. С тех пор ГА были успешно использованы для решения широкого круга задач, включая многие комбинаторные задачи оптимизации, оптимизацию многомерных функций, машинное обучение и эволюцию сложных структур, таких как нейронные сети. В работах [6, 9] Голдбергом и Михаливичем дается обзор ГА и приложений на их основе в различных сферах.

В природе генетическая информация определяется четверичным кодом, основанном на четырех нуклеотидах: аденине, цитозине, гуанине и тимине, соединенных вместе в последовательность ДНК, которая лежит в основе генетического кода [10]. При переносе этой структуры в информатику, кажется естественным основывать все кодирование на двоичном коде, общепринятом в компьютерных науках. То есть используются хромосомы, которые являются двоичными строками, кодирующими решения поставленных задач, и именно это является отличительной чертой генетических алгоритмов [11]. ГА выполняет глобальный поиск в пространстве решений, которое состоит из векторов данных. Первым шагом алгоритма является инициализация пространства решений случайно сгенерированными значениями. На каждой итерации, доступные решения мутируют или скрещиваются с другими решениями для того, чтобы создавать новые решения. В конце каждой итерации, каждый индивидуум (решение-кандидат) оценивается с использованием заданной фитнес- функции. Таким образом, фитнес-функция – ключевая часть каждого эволюционного алгоритма, и

предназначена для поиска того решения, которое является наилучшим для данной задачи. После того, как каждая особь была оценена, наименее пригодные кандидаты отклоняются, при этом остаются только лучшие из доступных решений в популяции. Цикл итераций повторяется до тех пор, пока предопределенный критерий останова не будет достигнут. В качестве критерия останова может использоваться либо число итераций, либо условие прохождения определенного порога значения фитнес-функции, которое должно быть достигнуто в пространстве решений. Более подробная информации по ГА может быть найдена в [6], [12], [13].

Рассмотрим, прежде всего, классический генетический алгоритм, являющийся базовым для эволюционных методов оптимизации. ГА используют принципы и терминологию, заимствованные у биологической науки – генетики. В ГА каждая особь представляет потенциальное решение некоторой проблемы. В классическом ГА особь кодируется строкой двоичных символов – хромосомой, каждый бит которой называется геном. Множество особей – потенциальных решений составляет популяцию. Поиск (суб)оптимального решения проблемы осуществляется в процессе эволюции популяции – последовательного преобразования одного конечного множества решений в другое с помощью генетических операторов репродукции, кроссинговера и мутации.

Генетические алгоритмы – не просто случайный поиск, они эффективно используют информацию, накопленную в процессе эволюции. ГА берет множество параметров оптимизационной проблемы и кодирует их последовательностями конечной длины в некотором конечном алфавите (в простейшем случае двоичный алфавит «0» и «1»). ГА работает до тех пор, пока не будет выполнено заданное количество поколений (итераций) процесса эволюции или на некоторой генерации будет получено заданное качество, или вследствие преждевременной сходимости при попадании в некоторый локальный оптимум.

В каждом поколении множество искусственных особей создается с использованием старых и добавлением новых с хорошими свойствами. В процессе поиска решения необходимо соблюдать баланс между «эксплуатацией» полученных на текущий момент лучших решений и расширением пространства поиска. Различные методы поиска решают эту проблему по-разному. Например, градиентные методы практически основаны только на использовании лучших текущих решений, что повышает скорость сходимости с одной стороны, но порождает проблему локальных экстремумов с другой стороны. В полярном подходе случайные методы поиска используют все пространство поиска, но имеют низкую скорость сходимости. В ГА предпринята попытка объединить преимущества этих двух противоположных подходов. При этом операторы репродукции и кроссинговера делают поиск направленным. Широту поиска обеспечивает то, что процесс ведется на множестве решений – популяции и используется оператор мутации.

В отличие от других методов оптимизации ГА оптимизируют различные области пространства решений одновременно и более приспособлены к нахождению новых областей с лучшими значениями целевой функции за счет объединения квазиоптимальных решений из разных популяций.

Классический ГА в общем случае содержит следующие шаги:

1. Создание начальной популяции.

1.1. Инициализация хромосомы каждой особи.

1.2. Оценивание начальной популяции.

2. Этап эволюции – построение нового поколения.

2.1. Отбор кандидатов на скрещивание (селекция).

2.2 Скрещивание, т.е. порождение каждой парой отобранных кандидатов новых индивидов.

2.3. Мутация.

2.4. Фильтрация хромосом, представляющих собой невозможные решения

2.5. Оценивание новой популяции.

Критерием останова алгоритма обычно являются либо прохождение максимально допустимого количества поколений, либо достижение функцией приспособленности какой-либо особи некоторого заранее заданного порогового значения.

2. Решение практической задачи

Было разработано вэб-приложение для построения расписания тренировок группы теннисистов разного пола, возраста и уровня игры. При построении расписания основным ограничением является время, в которое теннисист может принимать участие в тренировках.

Разработанное вэб-приложение состоит из двух частей:

– Frontend (интерфейс пользователя), использующий технологии HTML 5, CSS, Javascript, Ajax);

– Backend (непосредственно генетический алгоритм и логика работы приложения), использующий технологии Java 8, Struts-2 framework, Servlets.

Для хранения пользовательских данных (сведения о тренерах и теннисистах), вспомогательной служебной информации, а также о самих тренировках, использовался сервер mysql-5.7 (https://dev.mysql.com/). Приложение было размещено на локальном Веб-Сервере Apache Tomcat-8 (http://tomcat.apache.org/).

Главная страница разработанного приложения показана на рис. 1.

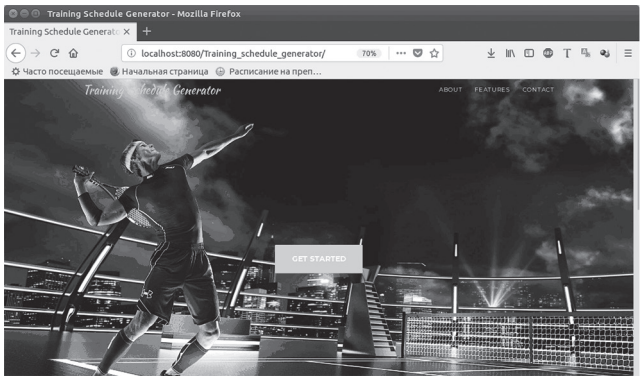

Рис. 1. Главная страница приложения

После нажатия на кнопку “Get Started” пользователь переходит на страницу Входа в систему/ Регистрации, вид которой приведен на рис.2. В данном приложении предполагается, что может быть несколько тренеров и у каждого из них будет своя отдельная учетная запись в системе. После успешной регистрации пользователь переходит на страницу успешной регистрации. Если же при регистрации или попытке входа в систему происходит какой-либо сбой, то пользователь переход на страницу повторного входа в систему.

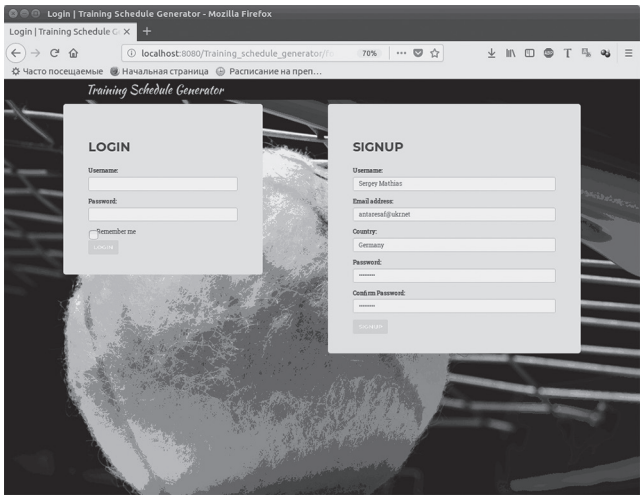

Рис. 2. Страница входа в систему и регистрации

После успешной регистрации и входа в систему пользователь переходит на страницу ввода данных, которая приведена на рис. 3.

В форме, показанной на рис.3, необходимо ввести информацию о количестве тренировок в день (рис. 4); о наличии перерывов между тренировками (рис. 5) и о том, по каким именно дням недели данные тренировки будут проходить; о тренерах и теннисистах (рис. 6).

После того, как вся информация была введена, пользователь может перейти непосредственно к составлению расписания. Пример построенного расписания показан на рис. 7. При составлении расписания генетическим алгоритмом учитывались следующие жесткие ограничения:

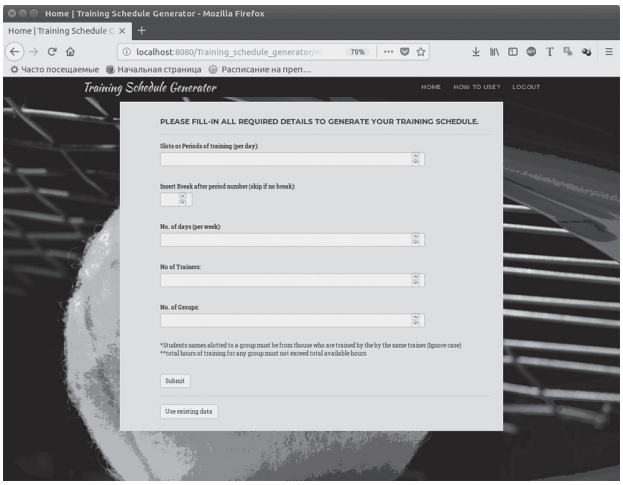

Рис. 3. Страница ввода данных

– максимальный размер группы – 4 человека;

– в группах могут быть игроки только одного пола;

– разница в возрасте игроков в одной группе не может превышать 3 лет;

– разница в силе удара игроков в одной группе не должна превышать 7 LK.

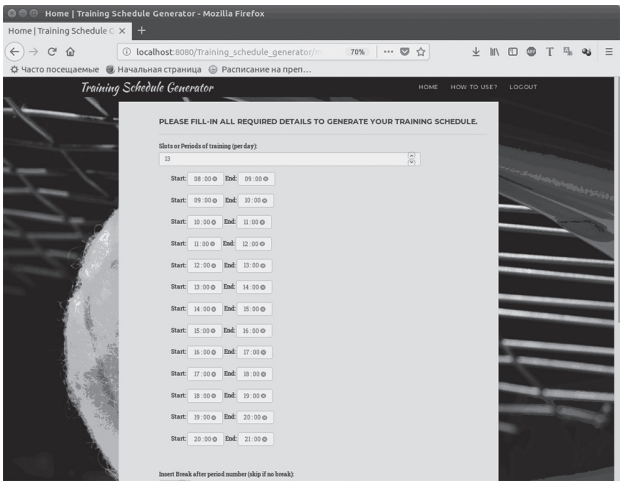

Рис. 4. Ввод информации о количестве тренировок в день

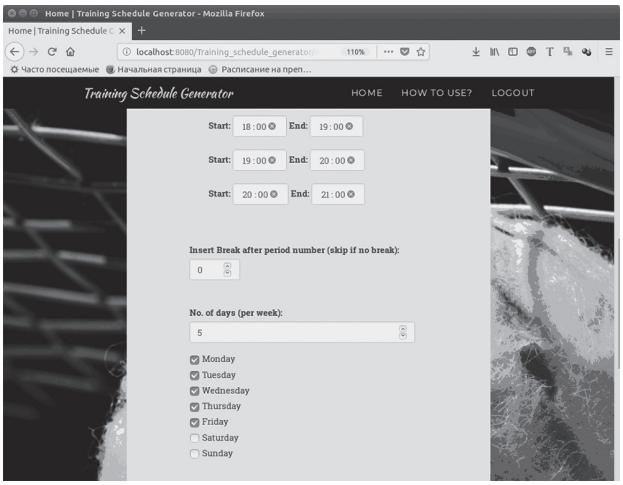

Рис. 5. Ввод информации о наличии перерывов между тренировками и о том, по каким именно дням недели тренировки будут проходить

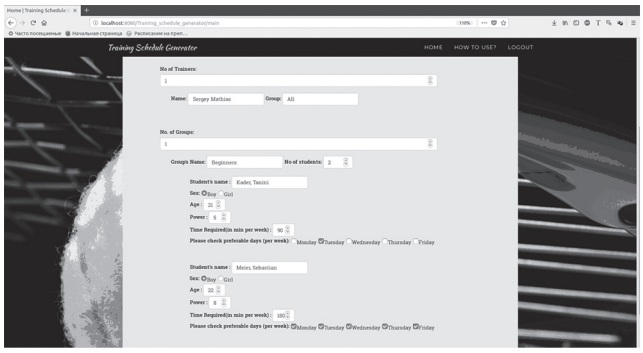

Рис. 6. Ввод информации о тренерах и студентах

3. Особенности реализации ГА в приложении

Инициализация популяции. В начале работы ГА производилась инициализация популяции, состоящей из 1000 особей. Каждая особь кодировалась с помощью хромосомы, представляющей из себя последовательность из 520 ген (26*5*4, где 26 – число слотов в день, 5 – количество игровых дней в неделе, 4 – максимальное количество игроков в период времени). Каждый ген являлся целым числом, соответствующим идентификационному номеру игрока в базе данных. Пример хромосомы, кодирующей одно из решений задачи составления расписания, показан на рис.8. Таким образом, при решении задачи использовалось прямое целочисленное кодирование.

При прямом кодировании само расписание является хромосомой. В этом случае расписание принимает форму последовательности событий, сгруппированных по датам и времени, либо список всех событий в расписании, каждое из которых имеет назначенное время. При таком кодировании представление решения в виде хромосомы обычно является довольно громоздким и требует разработки сложных генетических операторов для работы с ним. Так, например, оператору кроссовера необходимо обеспечить скрещивание таким образом, чтобы части двух разных расписаний при объединении формировали допустимый график. Также при изменении прямого представления расписания требуется оператор восстановления хромосомы, устраняющий ошибки, возникающие после применения генетических операторов. Однако в сложной реальной задаче, такой как составление расписания, прямое кодирование может быть наиболее предпочтительным, так как генетические операторы могут получить доступ к расписанию в его естественной форме и совершить эффективные перестановки и исправления.

В качестве альтернативы, расписание может быть закодировано в хромосоме непрямым образом, например, в виде перестановок игроков. При таком подходе вместо того чтобы кодировать всю информацию о расписании в самой хромосоме, список игроков служит в качестве основы для создания расписания. Порядок игроков в списке напрямую не отображает расписание, но он влияет на расписание, составленное на его основе. Затем это расписание может быть оценено на основе заданных ограничений. Недостатком косвенного кодирования является то, что трудно предсказать, как изменение порядка элементов в хромосоме повлияет на само расписание. Таким образом, задание конкретных изменений в расписании становится практически невозможным.

TRAINER: SERGEY RUDENKO

	Monday	Tuesday	Wendsday	Thursday	Friday
15:00-15:30					Tanja Schneider (2006, 23) Paulina Mayer (2006, 23) Julia Müller (2010, 23) Dejana Kovacevic (2006, 23)
15:30-16:00	**Sebastian Dörr (2000, 23) Jeremy Mayer (2002, 23) Heinze, Sven Niklas (2000, 22)**				Tanja Schneider (2006, 23) Paulina Mayer (2006, 23) Julia Müller (2010, 23) Dejana Kovacevic (2006, 23)
16:00-16:30	**Sebastian Dörr (2000, 23) Jeremy Mayer (2002, 23) Heinze, Sven Niklas (2000, 22)**		**Melvin Riemer (2009, 23) Daniel Magdic (2009, 23) Justin Becker (2007, 23) Ben Claus (2006, 23)**	Aylin Bergmann (2008, 23) Jekaterina Belaschova (2007, 23) Josefine Papenfuhs (2004, 23) Gloria Mann (2009, 23)	
16:30-17:00	**Sebastian Dörr (2000, 23) Jeremy Mayer (2002, 23) Heinze, Sven Niklas (2000, 22)**	**Marian, Stefan (1990, 4) Schad, Maximillian (1999, 16) Mannhart, Georg (1999, 18)**	**Melvin Riemer (2009, 23) Daniel Magdic (2009, 23) Justin Becker (2007, 23) Ben Claus (2006, 23)**	Aylin Bergmann (2008, 23) Jekaterina Belaschova (2007, 23) Josefine Papenfuhs (2004, 23) Gloria Mann (2009, 23)	**Faust, Henrik (2001, 16) Becker, Tim (2001, 23) Faust, Marlon (2002, 20) Steiner, Falk (2002, 18)**
17:00-17:30	**Florian Biehrer (2009, 23) Connor Lüning (2008, 23) Julian Pschorn (2007, 23)**	**Marian, Stefan (1990, 4) Schad, Maximillian (1999, 16) Mannhart, Georg (1999, 18)**	Elea Knaur (2004, 20) Timm Larah (2004, 21) Jelena Barun (2000, 20) Ionescu, Annabelle (2003, 21)	**Marian, Stefan (1990, 4) Holm, Steven (1998, 13) Schad, Maximillian (1999, 16)**	**Faust, Henrik (2001, 16) Becker, Tim (2001, 23) Faust, Marlon (2002, 20) Steiner, Falk (2002, 18)**
17:30-18:00	**Florian Biehrer (2009, 23) Connor Lüning (2008, 23) Julian Pschorn (2007, 23)**	**Marian, Stefan (1990, 4) Schad, Maximillian (1999, 16) Mannhart, Georg (1999, 18)**	Elea Knaur (2004, 20) Timm Larah (2004, 21) Jelena Barun (2000, 20) Ionescu, Annabelle (2003, 21)	**Marian, Stefan (1990, 4) Holm, Steven (1998, 13) Schad, Maximillian (1999, 16)**	**Faust, Henrik (2001, 16) Becker, Tim (2001, 23) Faust, Marlon (2002, 20) Steiner, Falk (2002, 18)**
18:00-18:30	Anna Kreis (2008, 23) Aylin Bergmann (2008, 23) Charlotte Wegner (2006, 23)	**Schautor Laurenz (2009, 23) Daniel Magdic (2009, 23) Leon (2004, 23) Connor Lüning (2008, 23)**	Elea Knaur (2004, 20) Jelena Barun (2000, 20) Ionescu, Annabelle (2003, 21)	**Marian, Stefan (1990, 4) Holm, Steven (1998, 13) Schad, Maximillian (1999, 16)**	
18:30-19:00	Anna Kreis (2008, 23) Aylin Bergmann (2008, 23) Charlotte Wegner (2006, 23)	**Schautor Laurenz (2009, 23) Daniel Magdic (2009, 23) Leon (2004, 23) Connor Lüning (2008, 23)**	**Max Grembowski (2002, 23) Sven Becker (2000, 23) Vincent Spreng (2002, 23) Henzler, Adrian (2002, 20)**	Jelena Barun (2000, 20) Papenfuhs, Antonia (2000, 19) Elea Knaur (2004, 20) Salzmann, Elisabeth (2000, 23)	Sondergeld, Rachel (1981, 23) Rosana Magdic (1980, 23) Müller, Katharina (1998, 18) Muerell, Carina (1992, 22)

Рис. 7. Пример расписания тренировок, полученного с помощью ГА

Рис. 8. Пример хромосомы, кодирующей решение задачи составления расписания

В результате проведенных исследований был сделан вывод о том, что применение прямого целочисленного кодирования является более простым для данной практической задачи. Однако при таком кодировании возникает необходимость реализации процедуры восстановления хромосом, которая исправляла бы (либо удаляла) из популяции невозможные (поврежденные) хромосомы, появляющиеся вследствие применения генетических операторов скрещивания и мутации. Выбор целочисленного (а не вещественного кодирования, которое более соответствует эволюционным стратегиям), обусловлен тем, что каждый ген хромосомы является порядковым номером игрока, поэтому целочисленное кодирование является в данном случае более предпочтительным.

Элитизм. Обычно при реализации ГА предполагается, что каждое новое поколение создается путем замены всех родителей их потомками. В соответствии с этой схемой репродукции генетический материал переходит к следующему поколению исключительно посредством применения операторов к выбранным родителям. Однако, хромосомы, выбранные с лучшими значениями пригодности, могут сочетаться неэффективным образом и создавать новые хромосомы с низкими значениями пригодности. Более того, стохастическая природа этих методов исключает какую-либо гарантию того, что сильные хромосомы вообще будут выбраны для процесса размножения. Таким образом, полезный генетический материал может быть утрачен для будущих поколений.

Элитизм позволяет устранить данную проблему путем автоматического копирования лучших хромосом из текущего поколения в новое. Остальная часть нового поколения создается с помощью схемы замены поколений. Основная проблема при таком подходе заключается в том, что элитные особи и многие идентичные или похожие их копии могут вскоре начать доминировать в популяции. Однако, как показали эксперименты, применение элитизма позволяет получить лучшие результаты по сравнению с ГА, не использующими элитарность. Поэтому при разработке программы составления тренировок была использована схема репродукции с использованием элитизма. При этом 10% лучших особей переходили (копировались) в новое поколение без каких-либо изменений их хромосом.

Оператор селекции. При разработке программы составления теннисных тренировок были протестированы различные операторы селекции. Наиболее эффективным оказался гибридный подход, который сочетает в себе случайную и ранговую селекцию. Данный оператор работает следующим образом: все особи популяции сортируются (ранжируются) в порядке убывания значения их фитнесс-функций. Максимальным значением фитнесс-функции для каждой особи является 1, а минимальным – 0. Затем датчиком случайных чисел генерируется некоторое число P, лежащее в интервале от 0 до N/10 (где N – размер популяции). После этого в цикле суммируются значения фитнесс-функций особей до тех пор, пока данная сумма не превышает P. Затем цикл останавливается и предпоследняя особь, участвовавшая в формировании общей суммы, выбирается в качестве одного из родителей. Такой подход позволяет обеспечить случайный выбор родителей из 10% наиболее приспособленных особей.

Оператор скрещивания. Основным оператором, влияющим на качество решений, полученных с помощью ГА, является оператор скрещивания [14]. Способность генетических алгоритмов совмещать различные решения отличает их от других эвристических методов. Поэтому операторы кроссовера должны быть реализованы таким образом, чтобы хорошо приспособленные родители могли быть интегрированы в одно более приспособленное потомство.

Наиболее часто в ГА применяется простой одноточечный оператор кроссовера. Он обменивает части родительских хромосом с каждой стороны случайно выбранной точки пересечения. Например, в двух 10-битовых хромосомах, если точка пересечения находится между пятым и шестым битами, первое потомство наследует от первого до пятого бита первого родителя и от шестого по десятый биты второго родителя. Второе потомство получает оставшиеся половинки родителей. Этот процесс очень похож на генетическую рекомбинацию в реальной ДНК. Одноточечный кроссовер эффективен в некоторых простых случаях, но в реальных задачах он обычно генерирует непригодные хромосомы [15]. Двухточечный же кроссовер обменивает гены вокруг двух точек пересечения, что позволяет составлять большее количество возможных комбинаций. Именно такой оператор и был использован при решении задачи составления расписания тренировок.

Оператор мутации. В то время как оператор кроссовера пытается объединить уже рассмотренные хорошие решения, оператор мутации вводит разнообразие в популяцию и позволяет алгоритму генерировать новые решения, которые невозможно получить лишь с использованием одного только оператора кроссовера. Кроссовер может привести только к созданию ограниченного набора

хромосом на основе родителей. Применяя небольшое количество мутаций к хромосоме, можно исследовать новые области поискового пространства.

Наиболее распространенным типом мутации с кодированием битовых строк является простая битовая мутация, в которой значение случайного бита в хромосоме родительской особи инвертируется. При использовании же целочисленного кодирования, мутация может быть реализована несколькими способами. Один метод, известный как позиционная мутация, состоит в удалении элемента из одной позиции и вставке его в хромосому в другую позицию, соответственно изменяя и другие гены. Другой подход, основанный на порядке следования генов, осуществляет перестановку двух элементов в хромосоме. Альтернативный подход, который считается весьма эффективным на практике, основан на скрамблировании генов в хромосоме. Этот метод создает хромосому потомка, переставляя гены родительской хромосомы.

Для каждого из операторов мутации необходимо определять параметр, регулирующий степень мутации. Для первых двух методов данный параметр определяет количество заменяемых или переставляемых генов, а для третьего подхода параметр мутации может определять размер области хромосомы-родителя, в которой будут осуществлены перестановки.

Вычисление фитнесс-функции. Как уже отмечалось, при решении реальной практической задачи с использованием ГА, построение эффективной фитнесс-функции является одним из самих сложных и ответственных этапов. Фитнесс-функция должна оценивать удовлетворение мягких и жестких ограничений, а также штрафовать за любое их нарушение. В разработанной программе, фитнесс-функция использует хромосому в качестве входных данных и возвращает значение пригодности в диапазоне от нуля (наименее приспособленная особь) до единицы (наиболее приспособленная особь).

В разработанном приложении фитнесс-функция представляла собой совокупность взвешенных шести рейтингов для каждого слота расписания, которые можно сформулировать следующим образом:

– R1 – рейтинг соответствия времени (игрок играет в то время, которое выбрал);

– R2 – рейтинг соответствия пола игроков;

– R3 – рейтинг соответствия возраста игроков;

– R4 – рейтинг соответствия силы удара игроков;

– R5 – рейтинг соответствия количества забронированных слотов выбранной продолжительности тренировки (60 мин – 2 слота подряд, 90 мин – 3 слота подряд);

– R6 – рейтинг компактности тренировок (используемые слоты находятся в непосредственной близости).

Тогда значение фитнесс-функции может быть вычислено следующим образом:

$$F(i) = \frac{\sum_{k=1}^{N}\sum_{l=1}^{6} w_l R_l(k)}{N},$$

где N — общее количество слотов; w_l — вес l-го рейтинга.

Максимальное значение данной функции равно 1, а минимальное – 0. Минимальное значение фитнесс-функции будет получено в случае, если для каждого слота будут нарушены все ограничения. Максимальное значение достигается в том случае, когда сумма всех рейтингов для каждого слота равна 1 (т.е. все ограничения удовлетворены). Тогда сумма рейтингов всех слотов будет равна N, а фитнесс-функция данного решения станет равной 1 (N/N), что для практической задачи обычно является недостижимым.

4. Сравнение решений, полученных с помощью ГА, алгоритма случайного поиска и разработанного экспертом-человеком

Одной из задач, которые часто возникают в исследованиях ГА, является демонстрация его превосходства над алгоритмом случайного поиска, в котором большое количество решений рассматривается случайным образом до тех пор, пока не будет найдено достаточно хорошее. В данной работе к сравнению было также добавлено решение, полученное экспертом-человеком. Для сравнения эффективности генетического алгоритма и алгоритма случайного поиска, оба алгоритма выполнялись в течении 60 мин при прочих равных условиях. Результаты эксперимента приведены на рис. 9.

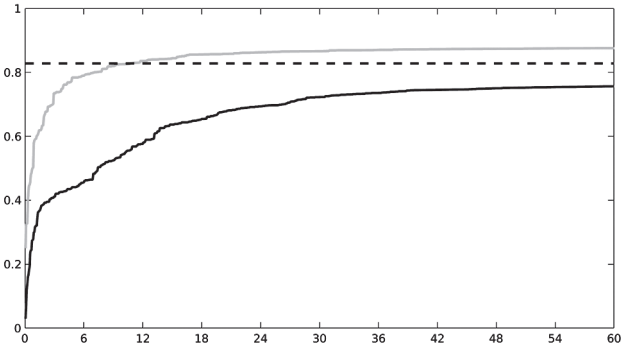

Рис. 9. Результаты сравнения ГА, алгоритма случайного поиска и решения, полученного экспертом-человеком

На рисунке красной пунктирной линией показано значение фитнесс-функции решения, полученного экспертом-человеком, зеленой линией показан график изменения фитнесс-функции наилучшего решения в популяции, полученного с помощью ГА, а синей – наилучшее решение,

полученное на текущий момент времени с помощью алгоритма случайного поиска.

Как видно из результатов эксперимента, эксперт-человек составил достаточно качественное расписание, которое превзошло все решения, полученные с помощью метода случайного поиска. Применение же генетического алгоритма позволило достаточно быстро найти решение, превосходящее по своим характеристикам решение эксперта. Следует, однако, отметить, что эксперт-человек обладает некоторыми преимуществами над автоматическими алгоритмами, так как может прибегать к сознательным незначительным нарушениям заданных ограничений с последующим их согласованием с теннисистами и тренерами. Автоматические же алгоритмы в свою очередь обладают мощным вычислительным потенциалом, недостижимым для человека. Эксперт физически не может разработать и оценить такое количество решений (порядка 200-300 тысяч), которое было рассмотрено алгоритмами.

Выводы

В данной статье рассмотрена задача составления расписания спортивных тренировок с помощью генетических алгоритмов, основным преимуществом которых является то, что многие параметры решаемой задачи могут быть закодированы в геноме и определяться параллельно. Более того, в отличие от большинства алгоритмов оптимизации, предназначенных для потактового решения задачи, ГА оперируют с множеством решений – популяцией, что позволяет достичь глобального экстремума, не застревая в локальных. При этом информация о каждой особи популяции кодируется в хромосоме (генотипе), а получение решения (фенотипа) осуществляется после эволюции (отбора, скрещивания, мутации) путем декодирования.

Результатом решения практической задачи является полная автоматизация процесса построения расписания тренировок для группы теннисистов. Результаты работы полученного вэб-приложения подтверждают высокую эффективность и быстродействие ГА.

Список литературы:

1. *Fogel D.B.* An introduction to simulated evolutionary optimization / D.B. Fogel // IEEE Trans. on Neural Networks. – 1994. – v.5. – №1. – P. 3-14. **2.** *Goldberg D.E.* Genetic algorithms and machine learning / D.E. Goldberg, J.H. Holland // Mach. Learn. – 1988. – v.3(2). – P. 95–99. **3.** *Mitchell T.M.* Machine learning / T.M. Mitchell. – McGraw-Hill, Boston. – 1997. – 267 p. **4.** *Holland J.H.* Adaption in Natural and Artificial Systems. University of Michigan Press, 1975. – 211 p. **5.** *Holland J.H.* Adaptation in Natural and Artificial Systems 2nd edn / J. Holland – Cambridge, MIT Press. – 1992. – 228 p. **6.** *Goldberg D.E.* Genetic Alogorithms in Search, Optimization and Machine Learning / D.E. Goldberg. – Addison Wesley Longman, 1989. – 372 p. **7.** *Tomassini M.* A survey of genetic algorithms / M. Tomassini // Annual Reviews of Computational Physics. – 1995. – Vol.3. – P. 87-118. **8.** *Dumitrescu D.* Evolutionary Computation / D. Dumitrescu, B. Lazzerini, L.C. Jain, A. Dumitrescu // CRC Press, 2000. **9.** *Michaliwicz Z.* Genetic Algorithms + Data Structurs = Evolution Programs / Z. Michaliwicz – Springer-Verlag Berlin Heidelberg New York, 3-rd ed., 1996. **10.** *Forrest S.* Genetic algorithms: Principles of natural selection applied to computation / S. Forrest // Science. – 1993. – v. 261(5123). – P. 872–878. **11.** *Grefenstette J.J.* Optimization of control parameters for genetic algorithms / J.J. Grefenstette // IEEE Transactions on Systems Man and Cybernetics.– 1986. – v. 16(1). – P. 122–128. **12.** *Mitchell M.* An Introduction to Genetic Algorithms (Complex Adaptive Systems) / M. Mitchell. – MIT Press, Cambridge, 1998. – 221 p. **13.** *Coley D.A.* An introduction to Genetic Algorithms for Scientists and Engineers / D.A. Coley – World Scientific Publishing, Singapore, 1999. – 244 p. **14.** *Daniel Costa.* An evolutionary tabu search algorithm and the NHL scheduling problem. INFOR, 88(8): 161 178, 1995. **15.** *George L.* Nemhauser and Michael A. Trick. Scheduling a major college basketball conference. Operations Research 46:1 8, 1998.

Поступила в редколлегию 05.04.2018

Рецензент: д-р техн. наук, проф. О.Г. Руденко, Харьковский национальный экономический университет имени С. Кузнеца, Харьков