УДК 159.688

Вестник СибГАУ
2014. № 5(57). С. 85–92

# ГИБРИДНЫЙ ЭВОЛЮЦИОННЫЙ АЛГОРИТМ АВТОМАТИЗИРОВАННОГО ФОРМИРОВАНИЯ ДЕРЕВЬЕВ ПРИНЯТИЯ РЕШЕНИЯ

Л. В. Липинский[1], Т. В. Кушнарева[1], Е. В. Дябкин[2], Е. А. Попов[1]

[1]Сибирский государственный аэрокосмический университет
Российская Федерация, 660014, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
E-mail: LipinskiyL@mail.ru, rare-avis@mail.ru, epopov@bmail.ru
[2]Красноярский государственный медицинский университет имени профессора В. Ф. Войно-Ясенецкого
Российская Федерация, 660022, г. Красноярск, ул. Партизана Железняка, 1
E-mail: dyabkyn@mail.ru

*Рассматриваются вопросы, связанные с построением и обучением деревьев принятия решения. Разбираются проблемы переобучения деревьев принятия решения и их способности к обобщению. Обсуждаются проблемы локальных поисковых процедур для настройки деревьев принятия решения. Предлагается эволюционный подход к автоматизированному формированию деревьев принятия решения на основе гибридизации алгоритма генетического программирования и генетического алгоритма. Алгоритм генетического программирования решает задачу структурного поиска в пространстве деревьев принятия решения. Объектом поиска для генетического программирования служит взаимное расположение функциональных узлов в дереве принятия решения (структура). Структура дерева выстраивается из элементов терминального и функционального множеств. В качестве элементов функционального множества выбираются возможные условия, которые ограничивают одну из входных переменных и разбивают исходную совокупность на группы. В качестве элементов терминального множества выбираются возможные решения, которые принимаются в рамках рассматриваемой задачи. Генетический алгоритм решает задачу параметрической оптимизации. Каждое условие в дереве имеет один или несколько числовых параметров. Эти параметры кодируются в бинарную строку и осуществляется поиск в направлении уменьшения ошибки дерева принятия решения на обучающей выборке. Приводится совместная схема работы генетического программирования и генетического алгоритма. Рассматриваются основные эволюционные операторы. Предложенный подход был реализован в виде программной системы, позволяющей получать бинарные деревья принятия решения с узлами, содержащими не более одного параметра. Приводятся рабочие окна программы. С помощью программной системы решена задача о диагностике степени тяжести повреждений органов брюшной полости при перитоните. Показано, что эволюционный алгоритм, построенный на основе гибридизации алгоритма генетического программирования и генетического алгоритма, позволяет получать деревья принятия решения с высоким свойством обобщения, использующие в 4 раза меньше информации в сравнении с теми данными, которые фиксируются при анамнезе пациента.*

*Ключевые слова: генетический алгоритм, генетическое программирование, деревья принятия решений.*

Vestnik SibGAU
2014, No. 5(57), P. 85–92

# HYBRID EVOLUTIONARY ALGORITHM FOR AUTOMATED DESIGN OF DECISION TREES

L. V. Lipinskiy[1], T. V. Kushnareva[1], E. V. Dyabkin[2], E. A. Popov[1]

[1] Siberian State Aerospace University named after academician M. F. Reshetnev
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660014, Russian Federation
E-mail: LipinskiyL@mail.ru, rare-avis@mail.ru, epopov@bmail.ru
[2] Krasnoyarsk State Medical University
47, Partizana Zheleznyaka Str., Krasnoyarsk, 660022, Russian Federation
E-mail: dyabkyn@mail.ru

*Issues related to the design and training of decision trees, are considered in the paper. Problems of decision trees over-fitting are discussed as well as their ability to the generalization. Questions of local search procedures for the adjustment of decision trees are described. An evolutionary approach to the automated design of decision trees is suggested based on a hybridization of the genetic programming and genetic algorithm. Genetic programming fulfills the search of effective structures in the space of decision trees. Genetic programming is searching for effective variant of decision trees functional nodes position and interconnections, i.e. a structure. The tree structure is built with elements of terminal and functional sets of genetic programming. Elements of the functional set are conditions which limited one*

*of input variables and divided original data set into subgroups. Elements of the terminal set are possible decisions which should be made within solving a problem in hand. Genetic algorithm solves the problem of parameters optimization. Each condition within a decision tree has one or more parameters which are coded into binary string. Genetic algorithm seeks parameters which minimize an error of the decision tree on the instances of the training sample. In the article, the framework of the joint execution of genetic programming and genetic algorithm is given and main evolutionary operators are considered. Suggested approach was implemented as a computing system that allows designing binary decision trees with one parameter nodes. Operation windows of the implemented computing system are presented. The problem of diagnosing the severity of injuries of the abdominal cavity with peritonitis has been solved with the use of developed approach. It was demonstrated that the evolutionary techniques based on the hybridization of the genetic programming and the genetic algorithm allow automated designing decision trees with the high ability to generalization which use four times less information comparing to data usually collected in a patient anamnesis.*

*Keywords: genetic programming, genetic algorithm, decision trees, design automation, medical diagnosis.*

**Введение.** Деревья принятия решения (ДПР) широко и успешно применяются в практике интеллектуального анализа данных, представлении знаний и принятии решения [1–3]. По своей форме ДПР близки к формальному рассуждению эксперта и интуитивно понятны пользователю (специалисту предметной области). ДПР представляют собой направленный граф. Конечные вершины графа (терминальные), т. е. вершины, не имеющие исходящих дуг, представляют собой выводы (альтернативы выбора). Промежуточные вершины графа (функциональные), т. е. вершины, имеющие исходящие дуги, представляют собой условия перехода. Эти условия определяют, по какой дуге графа осуществляется переход (рис. 1).

Процесс принятия решения начинается с некоторой исходной функциональной вершины. В соответствии с условиями перехода осуществляется проход графа до некоторой терминальной вершины, которая и содержит в себе ответ на исходный вопрос (рис. 2).

Для того, что бы ДПР имело смысл и могло быть применено на практике, его необходимо обучить. Обучение заключается в выборе и структуризации вершин, а также в настройке числовых параметров условий. Исследователю необходимо выбрать структуру дерева и настроить его параметры. Большинство подходов настройки являются локальными и руководствуются принципом «разделяй и властвуй». Выбирается некоторое исходное условие, которое всю совокупность входных объектов делит на две или более групп. Если для этих групп невозможно выполнить вывод, они, в свою очередь, делятся на подгруппы. Это деление выполняется до тех пор, пока каждой группе входных объектов, образованной условиями функциональных вершин, не будет сопоставлен некоторый выход. Сложность обучения связана с необходимостью обеспечить компромисс между точностью решения и обобщенностью получаемых правил.

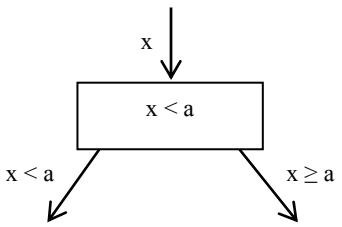


Рис. 1. Функциональный узел ДПР:
x – входящая переменная; a – параметр узла

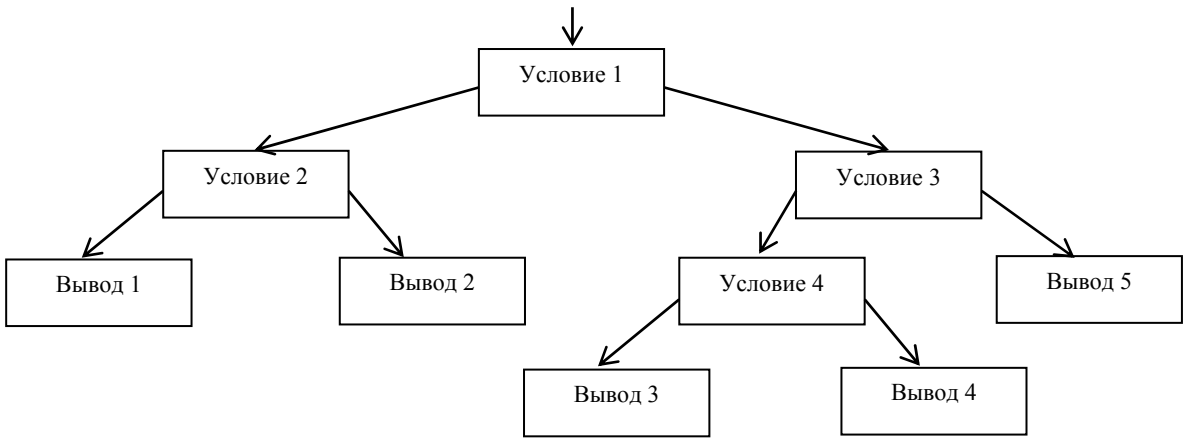


Рис. 2. Дерево принятия решения

Пусть мы получили абсолютно точное дерево, в котором все исходные объекты делятся на группы, где каждому объекту соответствует своя группа и каждой группе сопоставлен выход. В этом случае ДПР

«запомнило» обучающую выборку и не может быть применено к «неизвестным» объектам. Такое дерево равносильно самой выборке и не несет нового знания. Свойством обобщения называют свойство модели принимать правильные решения на тех данных, на которых не выполнялось обучение. Например, таким свойством обладают нейронные сети [4–6]. Модель не «запоминает» исходные данные, а отражает принципиальные зависимости между входами и выходами. В этом случае допускаются не абсолютно правильные решения, а близкие к правильным, достаточно близкие к правильным, хорошие решения. Компромисс между точностью и обобщенностью – это в том числе и компромисс, связанный с глубиной дерева. Относительно короткие деревья могут давать неточные, но хорошие решения. Деревья с большой глубиной скорее запоминают выборку, чем отражают природные зависимости между входами и выходами.

Локальные подходы к обучению ДПР нередко приводят к деревьям с большой глубиной и низким свойством обобщения. Неудачный выбор начальных условий приводит к необходимости увеличивать глубину дерева. Повысить эффективность поиска деревьев с высоким свойством обобщения может применение алгоритма генетического программирования (ГП) [7–9].

**Генетическое программирование.** ГП – это эволюционный метод поиска в пространстве иерархических структур. В отличие от генетического алгоритма (ГА) ГП оперирует иерархическими деревьями, что позволяет успешно настраивать сложные структурные объекты [10–12].

Для применения ГП необходимо решить две основные задачи: представить объекты поиска в виде иерархических деревьев и формализовать процедуру их оценки. В нашем случае первая задача решается достаточно легко. ДПР по своей природе являются иерархическими древовидными структурами и могут быть применены в ГП в исходном виде. Для решения второй задачи воспользуемся следующей формулой:

$$\text{fitness} = \frac{1}{1 + E + \alpha \cdot n}, \qquad (1)$$

где fitness – оценка качества ДПР; $E$ – ошибка ДПР, вычисляемая по формуле (2); α – коэффициент штрафа, накладываемого на глубину дерева; $n$ – величина, отражающая глубину или объем дерева. В качестве $n$ удобно выбирать либо количество вершин в дереве, либо саму глубину дерева:

$$E = \frac{1}{N}\sum_{i=1}^{N}(Y_i(X_i, K) - Y_i^*)^2 \xrightarrow[K]{} \text{opt}, \qquad (2)$$

где $Y_i^*$ – реальный $i$-й выход задачи, соответствующий $i$-му входу $X_i$; $K$ – набор числовых коэффициентов, используемых в дереве принятия решения; $Y_i(X_i, K)$ – выход, полученный по дереву решений для $K$ и $X_i$; $N$ – объем выборки.

Общий алгоритм работы ГП представлен на схеме (рис. 3).

Рассмотрим этапы алгоритма подробнее.

*Инициализация*. Здесь происходит создание стартовой популяции. Популяция состоит из индивидов, каждый из которых представляет собой некоторое ДПР. Для создания начальных индивидов необходимо определить функциональное и терминальное множества. Функциональное множество – это множество тех условий, которые могут быть включены в ДПР. Терминальное множество – это множество выходов (решений, альтернатив). На этапе инициализации деревья формируются случайным выбором вершин из терминального и функционального множеств.

*Оценка*. На этапе оценивания вычисляют пригодность. Пригодность – это количественная оценка того, насколько хорошо ДПР справляется со своей задачей. Исходная выборка разделяется на входы и выходы, входы подаются на ДПР, а вычисленные выходы дерева, т. е. результаты работы ДПР, подставляются в (1) для определения пригодности (fitness) для каждого индивида.

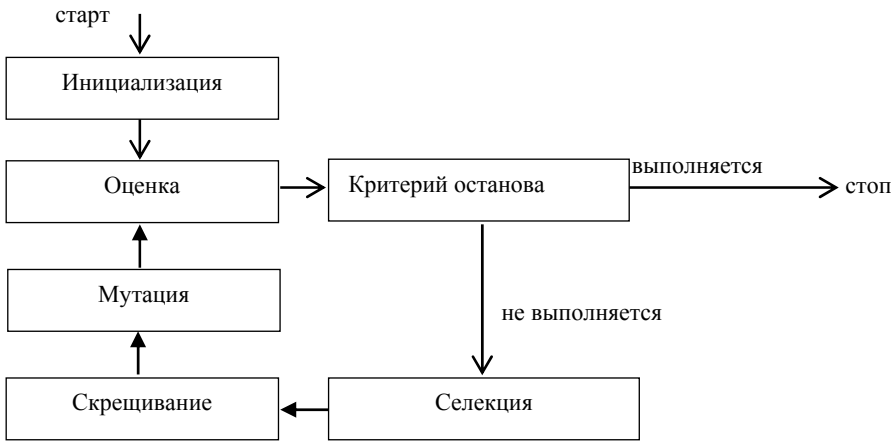


Рис. 3. Общая схема алгоритма работы ГП

*Селекция*. На этапе селекции отбираются родительские пары, которые будут использованы при скрещивании для порождения потомков, т. е. следующего поколения деревьев решений. Селекция оп-

ределяет направление поиска. Для того, чтобы поиск имел направленность, индивиды с большей пригодностью должны отбираться для скрещивания с большей вероятностью. Наиболее распространенные виды селекции – пропорциональная, ранговая, турнирная. При пропорциональной селекции вероятность скрещивания индивида пропорциональна его пригодности. При ранговой селекции вероятность скрещивания индивида пропорциональна его рангу. Например, если отсортировать индивидов по возрастанию пригодности, то рангом может быть порядковый номер индивида. При турнирной селекции отбирается индивид – победитель турнира. Случайным образом из популяции отбирается группа индивидов для турнира. Победителем турнира считается индивид с наибольшей пригодностью в группе.

*Скрещивание*. На этапе скрещивания родительские пары обмениваются генетическим материалом, в результате чего возникают потомки. Родительская пара образует пару потомков. Один из этих потомков (лучший по пригодности или отобранный случайным образом) передается в следующую популяцию. Скрещивание может быть одноточечным или стандартным. При стандартном скрещивании точка разрыва выбирается случайным образом у одного и другого родителя (рис. 4), при одноточечном скрещивании точка разрыва определяется общая для обоих родителей (рис. 5).

*Мутация*. На этапе мутации происходят случайные изменения в дереве с достаточно малой вероятностью. Функциональная вершина заменяется на случайно выбранную функциональную вершину, а терминальная вершина – на терминальную.

*Вычисление критерия останова*. Критерий останова определяет, в какой момент необходимо остановить алгоритм. Как правило, критерием останова является ограничение на вычислительный ресурс: количество вычисления целевой функции, количество итераций, время работы алгоритма и т. д. Результатом работы алгоритма является ДПР с наибольшей пригодностью.

Подробнее об эволюционных операторах можно узнать в [13–15].

**Гибридный эволюционный алгоритм автоматизированного проектирования ДПР.** При использовании генетического программирования для выбора эффективной структуры дерева принятия решений возникает необходимость оптимального выбора числовых параметров в функциональных узлах дерева. Для решения этого вопроса в общую схему метода включается генетический алгоритм оптимизации, настраивающий числовые параметры в функциональных узлах. Совместная работа алгоритмов показана на рис. 6.

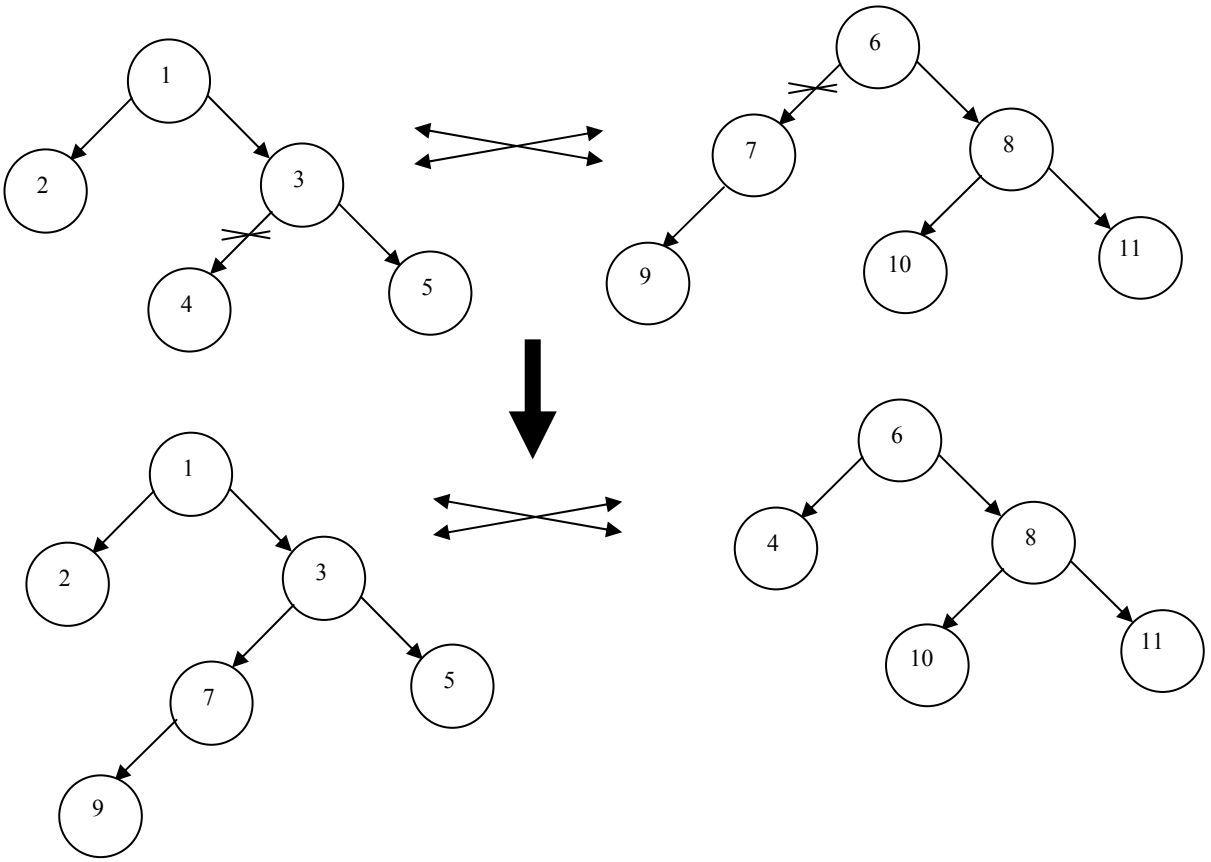


Рис. 4. Стандартное скрещивание

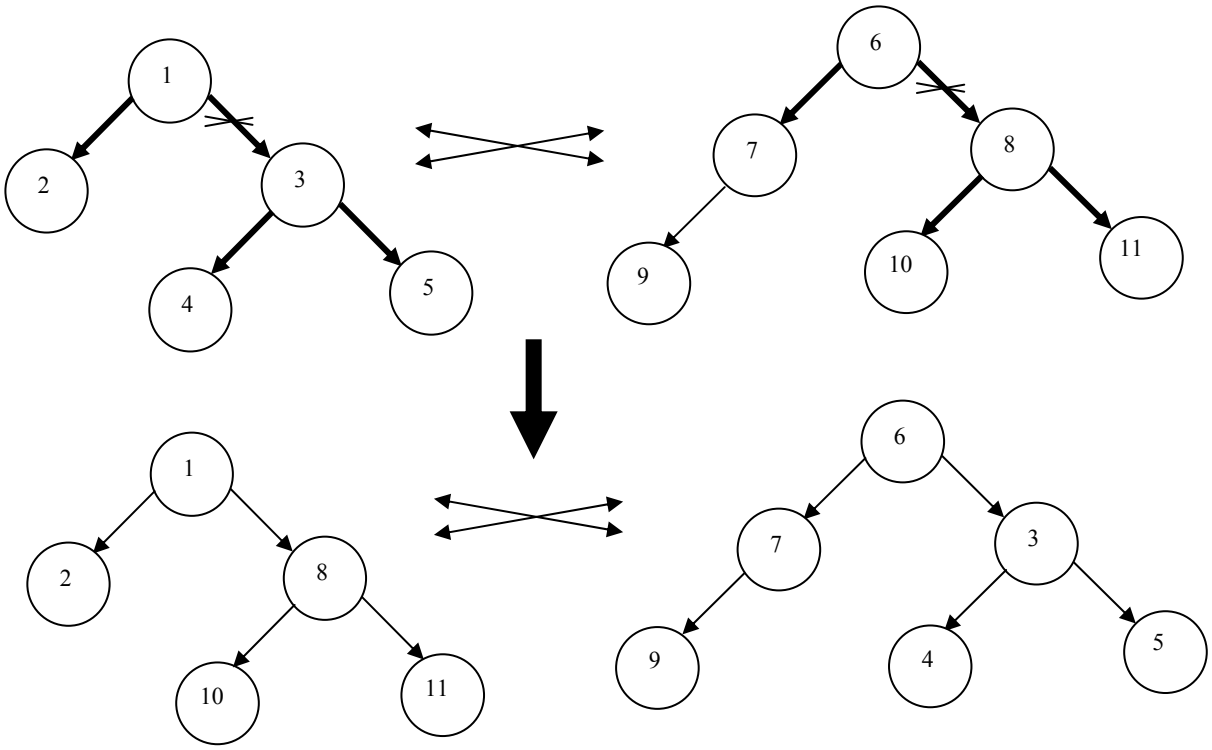


Рис. 5. Одноточечное скрещивание

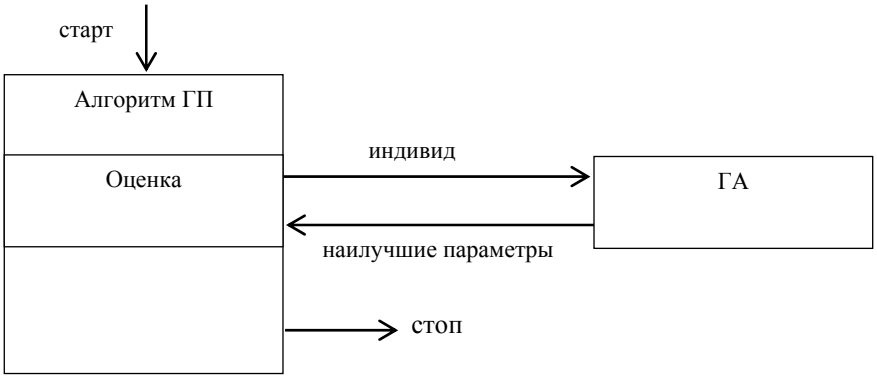


Рис. 6. Общая схема работы гибридного эволюционного алгоритма автоматизированного формирования ДПР

Описанный гибридный алгоритм был реализован в виде программной системы, рабочие окна которой представлены на рис. 7 и 8.

**Решение задачи медицинской диагностики.** После оценки работоспособности гибридного алгоритма на репрезентативном множестве тестовых задач было проведено исследование предложенного подхода на практической задаче диагностики степени тяжести поражения органов брюшной полости при перитоните. Для решения этой задачи была построена выборка, содержащая сведения о более чем ста пациентах. Для каждого пациента в выборке были определены следующие параметры.

Входные: 11 признаков гематологических интегральных показателей (ГИП), итоговая сумма баллов по показателям индекса брюшной полости, 1 признак – распределение по МПИ (мангеймский перитонеальный индекс) и 1 признак – итог по индексу брюшной полости.

Выходные: итоговая оценка степени тяжести, выполненная экспертом. Итоговая оценка содержит три градации: 1 степень, 2 степень и 3 степень, где 3 степень означает самые тяжелые повреждения.

Алгоритму были предоставлены достаточные вычислительные ресурсы. Для ГП: 10 индивидов, 10 поколений, максимальная глубина дерева 6, селекция турнирная, размер турнира 3, метод роста неполный, скрещивание одноточечное, мутация средняя. Для ГА: 10 индивидов, 10 поколений, селекция турнирная, размер турнира 3, средняя мутация, одноточечное скрещивание, точность поиска 0,01. Усредненные по 10 прогонам результаты работы алгоритма на тестовой и обучающей выборках представлены в табл. 1 и 2 соответственно. Пример дерева принятия решения представлен на рис. 9.

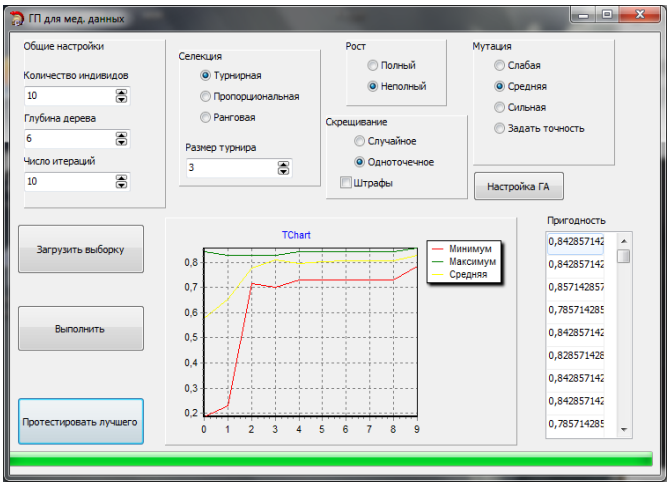


Рис. 7. Рабочее окно программной системы
(ГП для медицинских данных)

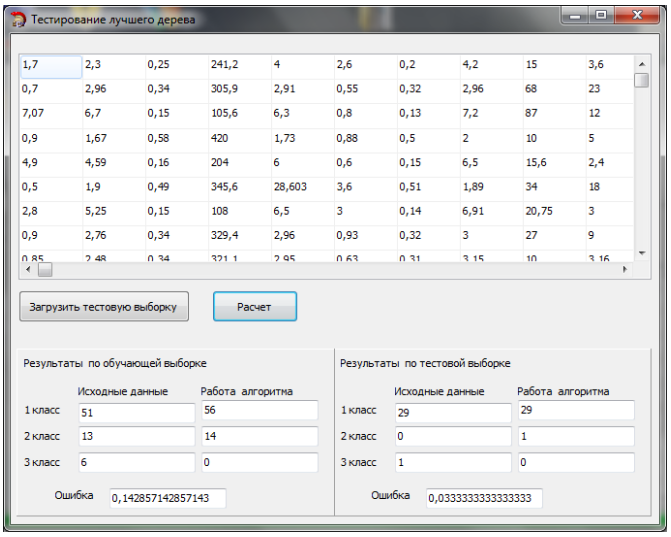


Рис. 8. Рабочее окно программной системы
(тестирование лучшего дерева)

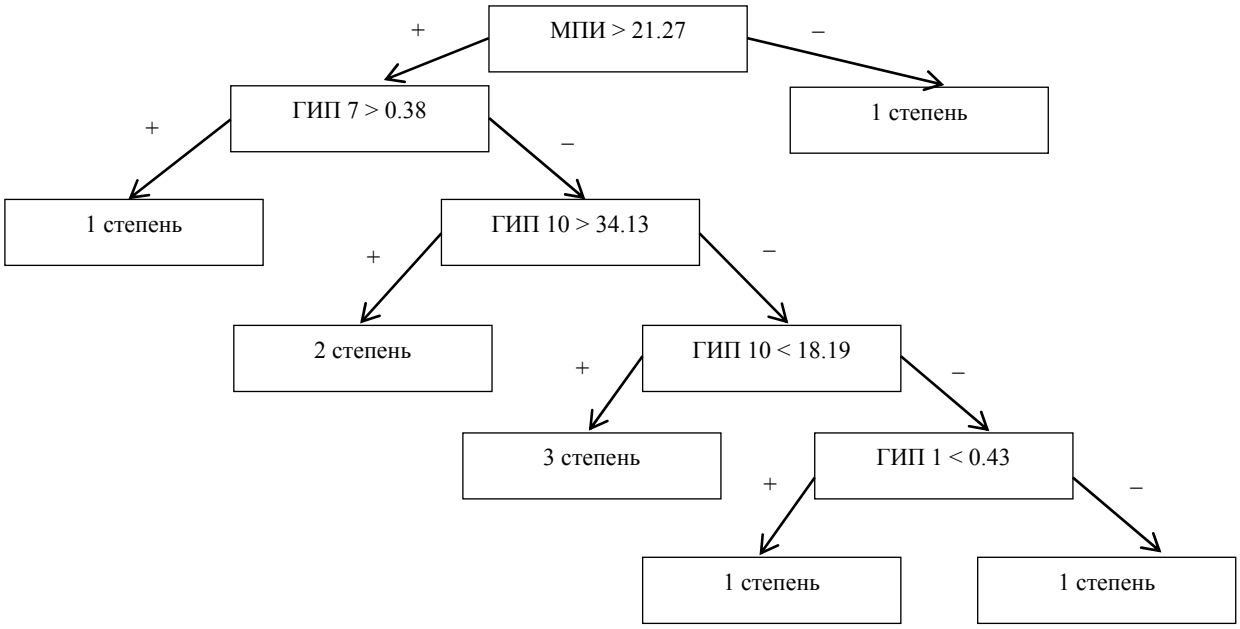


Рис. 9. Итоговое ДПР

*Таблица 1*

**Результаты работы алгоритма на тестовой выборке, усредненные по 10 прогонам**

| | Исходные данные | Определено верно | Определено ошибочно |
|---|---|---|---|
| Степень тяжести 1 | 29 | 29 | 0 |
| Степень тяжести 2 | 0 | 0 | 1 |
| Степень тяжести 3 | 1 | 0 | 0 |

*Таблица 2*

**Результаты работы алгоритма на обучающей выборке, усредненные по 10 прогонам**

| | Исходные данные | Определено верно | Определено ошибочно |
|---|---|---|---|
| Степень тяжести 1 | 51 | 51 | 5 |
| Степень тяжести 2 | 13 | 13 | 1 |
| Степень тяжести 3 | 6 | 0 | 0 |

**Заключение.** Анализ полученного дерева может быть произведен медицинским работником, не являющимся специалистом в интеллектуальных информационных технологиях. Например, нетрудно видеть, что из 13 исходных входов для принятия решения по полученному дереву необходимы только 4. Более того, на самом деле нужны только три из них, так как при любом значении ГИП 1 решение принимается одно и то же. То есть получаемые деревья решений строят существенно обобщенные решения, так как теперь для диагностирования требуется в четыре раза меньше информации, чем собирается по принятой в медицине практике. Можно видеть также, что для надежного диагностирования третьей степени заболевания необходимо собрать дополнительные данные, так как имеющихся примеров недостаточно для построения обобщенных правил диагностирования этой степени заболевания. Полученные данные представляют конкретный практический интерес для медицинских работников, помогают выявлять полезные закономерности и существенно повысить эффективность диагностики, а также создать системы поддержки принятия решения при диагностировании степени тяжести повреждений органов брюшной полости при перитоните.

Дальнейшее развитие подхода будет направлено на более полную автоматизацию проектирования деревьев решений путем исключения необходимости интуитивного выбора настроек используемых эволюционных алгоритмов за счет применения их самоконфигурирования [16; 17], так как это направление уже продемонстрировало свою эффективность в решении аналогичных прикладных задач [18]. В настоящий момент такой выбор настроек делается специалистом в области эволюционных вычислений и не может осуществляться конечным пользователем системы.

**Библиографические ссылки**

1. Quinlan J. Ross. Programs for Machine learning // Morgan Kaufmann Publishers. 1993. P. 302.

2. Методы и модели анализа данных: OLAP и Data Mining : учеб. пособие / А. А. Барсегян [и др.]. СПб. : БХВ-Петербург, 2004. 336 с. : ил.

3. Russel S., Norvig P. Artificial Intelligence: A Modern Approach. 3rd Edition. Pearson Education Limited, 2009. 1099 p.

4. Haykin S. Neural networks, a comprehensive foundation. N. Y. : Macmillan College Publishing Company. 1994.

5. Короткий С. Современные микропроцессоры // Нейронные сети: алгоритм обратного распространения : сб. ст. / сост. В. В. Корнеев, А. В. Киселев. 2-е изд. М., 2000.

6. Короткий С. Современные микропроцессоры // Нейронные сети: основные положения : сб. ст. / сост. В. В. Корнеев, А. В. Киселев. 2-е изд. М., 2000.

7. Семенкин Е. С., Липинский Л. В. О коэволюционном генетическом алгоритме автоматизированного проектирования системы управления на нечеткой логике // Автоматизация и современные технологии. 2006. № 11.

8. Семенкин Е. С. Липинский Л. В. Применение алгоритма генетического программирования в задачах автоматизации проектирования интеллектуальных информационных технологий // Вестник СибГАУ 2006. Вып. 3(10).

9. Семенкин Е. С. Липинский Л. В. Структурная адаптация нейронной сети методом генетического программирования // Исследование, разработка и применение высоких технологий в промышленности : сб. тр. II Междунар. науч.-практ. конф. СПб. : Изд-во Политехн. ун-та, 2006.

10. О генетическом программировании [Электронный ресурс]. URL: http://www.genetic-programming.com (дата обращения: 17.01.2015).

11. Koza J. R. Genetic programming tutorial. Morgan Kaufmann Publishers, 1994.

12. Wright A. Genetic algorithms for real parameter optimization // Foundations of Genetic Algorithms. San Mateo, CA : Morgan Kaufmann, 1991. Pp. 205–218.

13. Poli Riccardo. Exact Schema Theory for Genetic Programming and Variable-Length Genetic Algorithms with One-Point Crossover // Genetic Programming and Evolvable Machines. 2001. № 2.

14. Eiben A. E., Smith J. E. Introduction to Evolutionary Computing. New York : Springer-Verlag Berlin Heidelberg. 2003. 299 p.

15. Haupt R. L., Haupt S. E. Practical Genetic Algorithms. 2ed. Wiley, 2004. P. 261.

16. Semenkin E., Semenkina M. Self-configuring genetic algorithm with modified uniform crossover operator // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 7331 LNCS (PART 1). 2012. Pp. 414–421.

17. Semenkin E., Semenkina M. Self-Configuring Genetic Programming Algorithm with Modified Uniform Crossover // Proc. of the IEEE Congress on Evolutionary Computation (CEC). Brisbane (Australia). 2012. P. 3401–3406.

18. Semenkina M., Semenkin E. Hybrid self-configuring evolutionary algorithm for automated design of fuzzy classifier // Advances in Swarm Intelligence / Y. Tan, Y. Shi, C. A. C. Coello (Eds.). 2014. PT1. LNCS 8794. Pp. 310–317.

**References**

1. J. Ross Quinlan. Programs for Machine learning. *Morgan Kaufmann Publishers*. 1993. P. 302

2. Barsegyan A. A., Kupriyanov M. S., Stepanenko V.V., Holod N.N. *Metody i modeli analiza dannykh. OLAP i Data Mining* [Methods and models of data analysis: OLAP и Data Mining]. SPb., BKhV-Peterburg Publ., 2004, 336 p.

3. Russel S., Norvig P. Artificial Intelligence: A Modern Approach (3rd Edition). Pearson Education Limited, 2009, 1099 p.

4. Haykin S. Neural networks, a comprehensive foundation. N. Y. : Macmillan College Publishing Company. 1994.

5. Korotkiy S. *Sovremennye mikroprotsessory. Neyronnye seti: algoritm obratnogo rasprostraneniya.* [Modern microprocessors. Neural networks: back-propagation algorithm]. Moscow, 2000.

6. Korotkiy S. *Sovremennye mikroprotsessory. Neyronnye seti: osnovnye polozheniya.* [Modern microprocessors. Neural networks: basic statements]. Moscow, 2000.

7. Semenkin E. S. Lipinskiy L. V. [On co-evolutionary genetic algorithm for automated design of fuzzy logic system control] *Avtomatizatsiya i sovremennye tekhnologii.* 2006, no. 11.

8. Semenkin E. S. Lipinskiy L. V. [Application of genetic programming algorithm in problems of intelligence information technologies automated design] *Vestnik SibGAU*. 2006, no. 3 (10), p. 22–26 (In Russ.).

9. Semenkin E. S. Lipinskiy L. V. [The structural adaptation of neural network with genetic programming algorithm]. *Sb. tr. II Mezhdunar. nauch.prakt. konf. "Issledovanie, razrabotka i primenenie vysokikh tekhnologiy v promyshlennosti"* [Research, development and application of high technologies in the industry: Sat. tr. II International. scientific and practical. Conf.]. SPb. : Izd-vo Politekhn. un-ta Publ., 2006 (In Russ.).

10. John R. Koza. Genetic programming Inc. Available at: http://www.genetic-programming.com. (accessed 17.01.2015).

11. Koza J. R. Genetic programming tutorial. Morgan Kaufmann Publishers. 1994.

12. Wright A. Genetic algorithms for real parameter optimization. Foundations of Genetic Algorithms. San Mateo, CA: Morgan Kaufmann, 1991, p. 205–218.

13. Poli Riccardo. Exact Schema Theory for Genetic Programming and Variable-Length Genetic Algorithms with One-Point Crossover. Genetic Programming and Evolvable Machines, 2, 2001.

14. Eiben, A. E., Smith J. E. Introduction to Evolutionary Computing. Springer-Verlag Berlin Heidelberg New York, 2003. 299 p.

15. Haupt R. L., Haupt S. E. Practical Genetic Algorithms, 2ed., Wiley, 2004. P. 261.

16. Semenkin, E., Semenkina, M. Self-configuring genetic algorithm with modified uniform crossover operator, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 7331 LNCS (PART 1), 2012, p. 414–421.

17. Semenkin E., Semenkina M. Self-Configuring Genetic Programming Algorithm with Modified Uniform Crossover. *Proc. of the IEEE Congress on Evolutionary Computation*, (CEC), Brisbane (Australia). 2012. P. 3401–3406.

18. Semenkina M., Semenkin E. Hybrid self-configuring evolutionary algorithm for automated design of fuzzy classifier. In Y. Tan, Y. Shi, C.A.C. Coello (Eds.), *Advances in Swarm Intelligence*. 2014. PT1, LNCS 8794, P. 310–317.