УДК 519.6 DOI 10.23683/2311-3103-2017-3-140-148

И.Ш. Рудова, В.Г. Кобак

ИССЛЕДОВАНИЕ ВЛИЯНИЯ СИЛЬНЫХ МУТАЦИЙ ПРИ РЕШЕНИИ ЗАДАЧИ КОММИВОЯЖЕРА МОДИФИЦИРОВАННОЙ МОДЕЛЬЮ ГОЛДБЕРГА

Исследовано влияние сочетания различных кроссоверов и мутаций на результат работы генетического алгоритма (ГА) при решении задачи коммивояжера (ЗК). ЗК является NP сложной задачей дискретной оптимизации, для которой не найдено, а возможно и не существует алгоритма, позволяющего находить точное решение за полиномиальное время. Вероятность мутации и кроссовера в предложенном алгоритме составляет 100 %. Испытания проводились на графах средней размерности. ГА один из эффективных полиномиальных алгоритмов для нахождения приближённых решений ЗК. ГА относится к эволюционные методам решения задач, предназначенных для поиска предпочтительных решений и основаны на статистическом подходе к исследованию ситуаций и итерационным приближении к искомому состоянию систем. В использованной версии ГА применяется модифицированная модель Голдбегрга. В качестве стратегии отбора в ГА использовалась стратегия - «Сравнение мутирующей особи с предком, а далее лучшей из них со случайной особью в поколении». Она показала себя, как одна из лучших в ходе различных вычислительных экспериментов. Разработано программное обеспечение, реализующие данный алгоритм. Предложенный метод позволяет находить известное лучшее решение для графов до 30 вершин. А также отклонение лучшего найденного решения от оптимального не превышает 5.1 %, что является вполне допустимым для графов с размерностью от 50 до 60 вершин. Также следует отметить, что разница между лучшим и средним значениями из 30 запусков составляет приблизительно 1.5 %.

Задача коммивояжера; генетический алгоритм; граф; модель Голдберга; селекция, мутация; кроссовер; природные вычисления; гамильтонов цикл.

I.S. Rudova, V.G. Kobak

INVESTIGATION OF THE INFLUENCE OF STRONG MUTATIONS IN THE SOLUTION OF THE TRAVELING SALESMAN PROBLEM BY THE MODIFIED GOLDBERG MODEL

The present paper studies the influence of a combination of various crossovers and mutations on the result of genetic algorithm (GA) performance in solving the traveling salesman problem (TSP). TSP is an NP challenging discrete optimization problem, for which there is no algorithm found, and perhaps there is no algorithm which would allow finding the exact solution in polynomial time. The probability of a mutation and crossovers in the proposed algorithm is 100 %. The tests were carried out on average dimension graphs. GA is one of the effective polynomial algorithms for finding approximate solutions of TSP. GAs refer to the evolutionary methods of solving problems designed to find the preferred solutions and are based on the statistical approach to the study of situations and the iterative approximation to the sought-for state of systems. As a selection policy we used the "comparison of a mutating individual with an ancestor, and then the best of them with a random individual in the generation." It has shown itself as one of the best in the course of various computational experiments. The software implementing this algorithm was developed. This version of the GA uses a modified Goldberg model. The proposed algorithm allows us to find the known best solution for graphs up to 30 vertices. And also the deviation of the best solution found from the optimal does not exceed 5.1 %, which is quite acceptable for graphs with a dimension from 50 to 60 vertices. It should also be noted that the difference between the best and the average of 30 starts is about 1.5 %.

Traveling salesman problem; genetic algorithm; graph; the Goldberg model; selection; mutation; crossover; natural computing; hamiltonian cycle.

Введение. Задача коммивояжера в теории дискретной оптимизации считается классической. Впервые она была сформулирована еще в 1759 году [1]. Суть задачи состоит в том, чтобы найти кратчайший замкнутый путь обхода нескольких городов, заданных своими координатами (или с помощью матрицы расстояний между ними). Города могут посещаться только единожды. Очевидным решением задачи является полный перебор всех вариантов объезда и выбор оптимального по определенному критерию (время в пути, длина маршрута). Но если число городов равно n, то количество всевозможных маршрутов равно $n!$. Так как в задаче исходный пункт всегда известен заранее, остается перебрать $(n-1)\cdot(n-2)\cdot\ldots\cdot 2\cdot 1=(n-1)!$ возможных вариантов. Уже при 15 городах количество маршрутов достигает астрономических цифр [2]. Поиск точных и приближённых способов решения задачи о коммивояжёре остается актуальным и с теоретической и с практической точки зрения, так как не найдено, а возможно и не существует точных алгоритмов, решающих ЗК за короткое (полиномиальное) время. Поэтому для решения задачи коммивояжера эффективнее использовать не точные, а эволюционные методы [3]. Основное преимущество эволюционных методов оптимизации заключается в возможности решения, многомодальных (имеющих несколько локальных экстремумов) задач с большой размерностью за счет сочетания элементов случайности и детерминированности точно так же, как это происходит в природной среде[4].

Детерминированность этих методов заключается в моделировании природных процессов отбора, размножения и наследования, происходящих по строго определенным правилам, основным из которых является закон эволюции: «выживает сильнейший». Другим важным фактором эффективности эволюционных вычислений является моделирование процессов размножения и наследования. Рассматриваемые варианты решений могут по определенному правилу порождать новые решения, которые будут наследовать лучшие черты своих «предков»[5].

Постановка задачи. Задача коммивояжера (ЗК) NP-полная (задача с экспоненциальной оценкой числа итераций, необходимых для отыскания точного решения) и мультимодальная (имеет локальные экстремумы). Определяется она следующим образом: для заданного полного взвешенного графа G=(V,E,D) с множеством вершин $V=\{v_i\}_n$, множеством ребер E и матрицей весов $D=\{d_{i,j}\}_{nxn}$ необходимо найти гамильтонов цикл $(v_{i1}, v_{i2},\ldots,v_{in})$, где $(i_1,\ldots,i_n)$ – перестановка на множестве {1,2,…,n} , а $v_i \in V$, с минимальной суммой весов дуг [6]:

$$\sum_{j=1}^{n-1} d_{i_j i_{j+1}} + d_{n_1} \to \min .$$

Вершины графа G часто называются городами, а любой гамильтонов цикл называется маршрутом коммивояжера [7].

Основные алгоритмы. В работе для решения ЗК используется генетический алгоритм (ГА) с модифицированной моделью Голдберга [8]. Идея генетических алгоритмов заимствована у живой природы и состоит в организации эволюционного процесса, конечной целью которого является получение решения в сложной задаче оптимизации.

Общая схема такого алгоритма может быть записана следующим образом [9].

1. Формирование начальной популяции.
2. Оценка особей популяции.
3. Отбор (селекция).
4. Скрещивание (оператор кроссинговера).
5. Мутация.
6. Формирование новой популяции.
7. Если популяция не сошлась, то возвращение к пункту 2. Иначе – конец алгоритма.

Оператор кроссинговера (ОК) также называемый кроссовером, является основным генетическим оператором, за счет которого производится обмен генетическим материалом между особями. Моделирует процесс скрещивания особей [10].

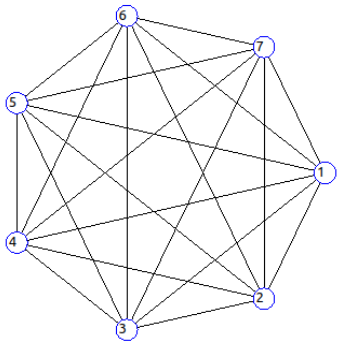
В данной работе используются 2 вида кроссоверов [11]: измененный ОК и двухточечный ОК. Работа измененного ОК может быть описана следующим образом: выбираются две родительские особи и точка сечения, например P1="1 2 3 6 | 4 5 7 1" и P2 = "1 2 6 4 | 7 5 3 1 ", затем первая часть одного родителя копируется в первого потомка – C1="1 2 3 6 | _ _ _ 1" . Вторая часть потомка заполняется генами второго родителя – C1= "1 2 3 6 | 7 5 _ 1". Если такие гены уже встречаются в потомке, то они пропускаются, а оставшуюся часть потомка дополняют гены первого родителя – C1= "1 2 3 6 | 7 5 4 1". Для второго потомка алгоритм аналогичен [12].

Для двухточечного ОК необходимо выбрать две родительские особи и две точки сечения P1="1 2 3 |6 4| 5 7 1" и P2 = "1 2 6| 4 7 | 5 3 1 ", каждый ген из центральной части первого родителя копируется в потомка, начальная и конечная точка сохраняются C1="1 _ _ |6 4| _ _ 1". Пробелы в потомке заполнятся генами второго родителя – C1="1 2 7 |6 4| 5 3 1".

Оператор мутации (ОМ) защищает популяцию от преждевременной сходимости. Были выбраны часто употребляемые сильные мутации (изменяется более 30% от генотипа особи) [13].

Жадная мутация – выбираются две точки сечения в хромосоме так, чтобы между ними было по крайне мере 2 города – P1 = "1 2 | 6 4 7 | 5 3 1 ", среди городов 6, 4, 7 выбирается ближайший к городу 2, предположим это город 4, он добавляется в хромосому сразу после второго города – M1= "1 2 | 4 _ _ | 5 3 1 ", далее среди городов 6 и 7 выбираем ближайший к городу 4 и т.д. [14].

Глобальная мутация – выбирается одна точка сечения в хромосоме – P1 = "1 2 6 4 | 7 5 3 1 ", далее все гены справа и слева от нее меняются местами – M1 = "1 7 5 3 | 2 6 4 1 "[15].



Ребро	::	Вес
1 2	::	17
1 3	::	6
1 4	::	14
1 5	::	13
1 6	::	16
1 7	::	1
2 3	::	1
2 4	::	14
2 5	::	11
2 6	::	9
2 7	::	4
3 4	::	15
3 5	::	15
3 6	::	8
3 7	::	5
4 5	::	10
4 6	::	9
4 7	::	16
5 6	::	12
5 7	::	11
6 7	::	9

Пример работы алгоритма. Рассмотрим граф, состоящий из 7 вершин:

Для тестового прогона алгоритма были выбраны следующие параметры: количество особей в поколении – 10, вероятность кроссовера и мутации – 100 %. Минимум должен быть неизменен не менее 10 итераций алгоритма подряд.

1. Вначале формируем начальное поколение, каждая особь формируется случайным образом. Далее задаем начальные значения для глобального минимума и счетчика итераций GlobalMin=INT_MAX, iter_index = 0.

```
# 0 :        "1 5 4 3 2 7 6 1  [68]"
# 1 :        "1 3 7 4 5 2 6 1  [73]"
# 2 :        "1 5 2 3 6 7 4 1  [72]"
# 3 :        "1 2 3 6 4 5 7 1  [57]"
# 4 :        "1 4 2 7 6 5 3 1  [74]"
# 5 :        "1 4 5 2 3 6 7 1  [54]"
# 6 :        "1 4 6 3 7 5 2 1  [75]"
# 7 :        "1 4 6 2 7 5 3 1  [68]"
# 8 :        "1 4 5 2 6 3 7 1  [58]"
# 9 :        "1 2 7 4 6 5 3 1  [79]"
```

2. i=0 – индекс текущей особи в поколении.

3. Формируем пару для кроссовера. Первый родитель – это i-ая особь, вторая родительская особь выбирается случайно. К примеру, случайным образом выбрана 4-я особь.

4. Кроссовер. Случайным образом выбираем точку разрыва. SplitPoint= 3.

parent1 "1 5 4 |3 2 7 6 1 [68]"

parent2 "1 4 2 |7 6 5 3 1 [74]"

Для тестового примера был выбран измененный одноточечный кроссовер. Первая часть одного родителя копируется в первого потомка, во вторую часть потомка копируются гены второго родителя. Если какие-то гены уже встречаются в потомке, то они пропускаются, а оставшаяся часть потомка заполняется генами первого родителя. Формирование второго потомка происходит аналогично.

child1 "1 5 4 7 6 3 2 1 [74]"

child2 "1 4 2 3 7 6 5 1 [68]"

5. Далее оба потомка подвергаются мутации, которая состоит в упорядочивании последовательности городов. Сначала случайным образом выбираются две точки сечения в хромосоме так, чтобы между ними было по крайне мере 2 города. Затем последовательность городов между точками сечения упорядочивается: они переставляются в зависимости от близости друг к другу.

Точки сечения – 2 и 6

child1 "1 5 |4 7 6 3| 2 1 [74]"

child1 после мутации = "1 5 |4 6 3 7| 2 1 [66]"

child2 "1 4 |2 3 7 6 |5 1 [68]"

child2 после мутации = "1 4| 6 3 2 7 |5 1 [60]"

Из двух потомков после оператора мутации, выбирается тот, чье значение ЦФ меньше, в нашем случае это child2

6. Отбор: значение целевой функции потомка сравнивается со значением целевой функции родителя, а затем особь с наименьшим значение целевой функции сравнивается со случайной особью из поколения[16,17]. Особь с наименьшим значением целевой функции добавляется в следующее поколение.

(parentЦФ=68) > (childЦФ=60)

Случайным образом выбираем особь из поколения – особь №8, её значение ЦФ = 75 >60, следовательно, в следующее поколение отправляется потомок.

7. Увеличиваем индекс i=i+1, i<числа особей в поколении? если да, то возвращаемся на шаг 3, если нет, то переходим на шаг 8.

8. Находим особь с наименьшим значением ЦФ в поколении, запоминаем его в переменную temp_min. Далее следует проверка не изменился ли глобальный минимум? Если temp_min< GlobalMin, то GlobalMin = temp_min, если нет то глобальный минимум не изменился, поэтому увеличиваем счетчик итераций на единицу iter_index=iter_index+1;

9. Если минимум не изменился заданное число раз (iteration_limit), то алгоритм заканчивает свою работу, если же iter_index < iteration_limit, то возвращаемся на шаг 2 [18].

Поколение на второй итерации:

```
# 0 :        "1 4 6 3 2 7 5 1  [60]"
# 1 :        "1 5 4 6 3 2 7 1  [46]"
# 2 :        "1 5 2 3 7 6 4 1  [62]"
# 3 :        "1 7 2 3 6 4 5 1  [46]"
# 4 :        "1 4 6 2 7 5 3 1  [68]"
# 5 :        "1 4 5 2 3 6 7 1  [54]"
# 6 :        "1 4 2 3 7 6 5 1  [68]"
# 7 :        "1 7 2 6 4 5 3 1  [54]"
# 8 :        "1 3 2 6 4 5 7 1  [47]"
# 9 :        "1 3 2 6 4 5 7 1  [47]"
```

Поколение на заключительной – десятой итерации тестового примера:

```
# 0 :        "1 5 4 6 3 2 7 1  [46]"
# 1 :        "1 5 4 6 3 2 7 1  [46]"
# 2 :        "1 5 4 6 3 2 7 1  [46]"
# 3 :        "1 5 4 6 3 2 7 1  [46]"
# 4 :        "1 5 4 6 3 2 7 1  [46]"
# 5 :        "1 5 4 6 3 2 7 1  [46]"
# 6 :        "1 5 4 6 3 2 7 1  [46]"
# 7 :        "1 5 4 6 3 2 7 1  [46]"
# 8 :        "1 5 4 6 3 2 7 1  [46]"
# 9 :        "1 5 4 6 3 2 7 1  [46]"
```

Минимальное значение ЦФ – 46.

Вычислительный эксперимент. Аналитически, сделать вывод об эффективности алгоритма нельзя, поэтому был поставлен вычислительный эксперимент. Эксперимент проводился на шести графах различных размерностей из пакета TSP_LIB Гейдельбергского университета, Гейдельберг, Германия [19]. В вычислительном эксперименте использовались различные комбинации кроссоверов и операторов мутаций. Были использованы следующие значения параметров алгоритма: количество особей – 500, условие останова (сколько раз не меняется минимум) – 250, вероятность кроссовера и мутации – 100%, количество запусков – 30. Результаты эксперимента представлены в табл. 1 и 2.

Для реализации алгоритма[20] использовалась среда QT Creator, framework QT 5.8.0 -1 и язык программирования C++. Эксперименты проводились на ЭВМ с операционной системой Arch Linux 6.3.1, процессором Intel 2.14 Ггц и оперативной памятью 4 Гб.

Таблица 1

Результаты вычислительного эксперимента для графов размерности до 30 вершин

	Граф fri26			Граф baeg29			Граф baes29		
	% откл ср. знач.	% откл лучшего знач.	Ср. время (мили сек.)	% откл ср. знач.	% откл лучшего. знач.	Ср. время (мили сек.)	% откл ср. знач.	% откл лучше-го. знач.	Ср. время (мили сек.)
Двуточечный кроссовер + глобальный ОМ	14,497	**2,348**	4879	10,787	**3,354**	4727,1	13,995	**3,564**	5263
Двуточечный кроссовер + жадный ОМ	0,082	**0,000**	16063	1,530	**0,000**	20151,3	1,295	**0,000**	19478
Измененный кроссовер + глобальный ОМ	20,135	**3,629**	13442	13,180	**5,155**	15095,5	13,508	**4,257**	14847
Измененный кроссовер + жадный ОМ	1,330	**0,000**	22218	3,176	**1,180**	25629,5	2,185	**0,000**	24503

Таблица 2

Результаты вычислительного эксперимента для графов размерности до 60 вершин

	Граф gr48			Граф berlin52			Граф brazil58		
	% откл ср. знач.	% откл лучшего знач.	Ср. время (мили сек.)	% откл ср. знач.	% откл лучшего. знач.	Ср. время (мили сек.)	% откл ср. знач.	% откл лучшего. знач.	Ср. время (мили сек.)
Двуточечный кроссовер + глобальный ОМ	27,550	**9,572**	18886,2	29,394	**15,911**	29271,5	39,459	**27,147**	41783,2
Двуточечный кроссовер + жадный ОМ	9,671	**5,648**	77244,4	6,050	**5,105**	109834	6,149	**3,875**	162336
Измененный кроссовер + глобальный ОМ	45,750	**18,193**	22399,8	33,743	**24,317**	30327,8	65,782	**46,060**	32406,3
Измененный кроссовер + жадный ОМ	8,348	**5,054**	74778,1	4,968	**3,328**	104202	6,991	**5,060**	132164

Выводы. Результаты вычислительного эксперимента демонстрируют эффективность использования модифицированной модели Голдберга для решения задачи коммивояжера. Самая лучшая комбинация упорядоченного кроссовера и жадного оператора мутации позволяет находить известное лучшее для графов, размерностью до 30 вершин. А для графов berlin52 и brazil58, отклонения лучшего результата составляют приблизительно 3.3 % и 5.06 % соответственно, что является вполне допустимым. Также следует отметить, что разница между лучшим и средним значениями из 30 запусков составляет приблизительно 1.5 %.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Задача коммивояжёра. Википедия. – URL: https://ru.wikipedia.org/Задача_коммивояжёра (дата обращения: 10.02.2017).
2. *Vairamuthu M., Porselvi S., DR. A. N. Balaji, J. Rajesh Babu.* Artificial Immune System algorithm for multi objective flow shop scheduling problem // International Journal of Innovative Research in Science, Engineering and Technology. – K.L.N. College of Engineering and Technology, Madurai, Tamil Nadu, India, March 2014. – No. 3. – P. 1391-1395.

3. *Мудров В.И.* Задача о коммивояжере. – М.: Либроком, 2013. – 16 с.
4. Why we are transposing or reversing the directions of all arcs (edges) in the Kosaraju two pass algorithm? – Режим доступа: https://www.quora.com/Why-we-are-transposing-or-reversing-the-directionsof-all-arcs-edges-in-the-Kosaraju-two-pass-algorithm (дата обращения: 02.03.2017).
5. *Kernighan B.W., Lin S.* An efficient heuristic procedure for partitioning graphs. – Режим доступа: https://www.cs.princeton.edu/~bwk/btl.mirror/new/partitioning.pdf (дата обращения: 02.03.2017).
6. *Matthias M., Fikret E.* Genetic Algorithms For Vertex Splitting in DAGs. – Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.8018&rep=rep1&type=pdf (дата обращения: 15.03.2017).
7. *Abdel K.I.* Path Partition in Directed Graph-Modeling and Optimization // New Trends in Mathematical Sciences. – Faculty of Arts and Sciences, Islamic University of Lebanon, 2013. – No. 1. – P. 74-84.
8. *Alamdari S., Mehrabian A.* On a DAG Partitioning Problem. – Режим доступа: http://www.math.uwaterloo.ca/~amehrabi/Articles/DAG_Partitioning.pdf (дата обращения: 23.03.2017).
9. *Гладков Л.А., Курейчик В.В., Курейчик В.М.* Генетические алгоритмы. – М.: Физматлит, 2006. – 320 с.
10. *Емельянов В.В., Курейчик В.М., Курейчик В.В.* Теория и практика эволюционного моделирования. – М.: Физматлит, 2003. – 432 с.
11. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы: построение и анализ. – М.: Вильямс, 2013. – 1328 с.
12. *Каширина И.Л.* Введение в эволюционное моделирование: учебное пособие / под ред. Е.С. Котляровой. – Воронеж: Изд-во ВГУ, 2007. – 40 с.
13. *Кобак В. Г., Кавтарадзе И.Ш., Бормотов В.В., Швидченко С.А.* Решение задачи коммивояжера модифицированной моделью Гольдберга с помощью различного вида мутаций // Тр. Северо-Кавказского филиала Московского техн. ун-та связи и информатики: междунар. молодеж. науч.-практ. конф. Т. 1. – Ростов-на-Дону, 2014. – С. 258-261.
14. *Кобак В.Г., Рудова И.Ш.* Возможности использования элитных особей при решении задачи коммивояжера моделью Гольдберга // Вестник компьютерных и информационных технологий. – 2016. – № 11 (149). – С. 3-7.
15. *Кобак В.Г., Рудова И.Ш., Жуковский А.Г.* Сравнительный анализ модифицированной модели Гольдберга и муравьиного алгоритма при решении задачи коммивояжера // Тр. Северо-Кавказского филиала Московского техн. ун-та связи и информатики: междунар. молодеж. науч.-практ. конф. Т. 1. – Ростов-на-Дону, 2015. – С. 362-365.
16. *Кобак В.Г., Рудова И.Ш.* Возможности использования элитных особей при решении задачи коммивояжера моделью Гольдберга // Вестник Донского государственного технического университета. – 2016. – Т. 16, № 2 (85). – С. 129-135.
17. *Кобак В.Г., Рудова И.Ш.* Решение задачи коммивояжера модифицированной моделью Гольдберга с использованием различных сильных мутаций // Сб. тр. Юбилейной конф. студентов и молодых ученых, посвященной 85-летию ДГТУ. – Ростов-на-Дону: ДГТУ, 2015. – С. 146-156.
18. *Кобак В.Г., Рудова И.Ш., Жуковский А.Г., Швидченко С.А.* Использование сильной мутации при решении задачи коммивояжера // Современные тенденции развития науки и технологий. – 2016. – № 6-1.
19. Решение задачи коммивояжера модифицированной моделью Голденберга с помощью совместного использования муравьиного и генетического алгоритмов: свидетельство о государственной регистрации программ для ЭВМ / Рудова И.Ш., Кобак В.Г. – 2016610345; дата регистрации 11.01.16 г.
20. TSP_LIB. – URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/ (дата обращения 20.02.17).

REFERENCES

1. Zadacha kommivoyazhera. Vikipediya [The traveling salesman problem. Wikipedia]. Available at: https://ru.wikipedia.org/Zadacha_kommivoyazhera (Accessed 10 February 2017).
2. *Vairamuthu M., Porselvi S., DR. A. N. Balaji, J. Rajesh Babu.* Artificial Immune System algorithm for multi objective flow shop scheduling problem, *International Journal of Innovative Research in Science, Engineering and Technology*. K.L.N. College of Engineering and Technology, Madurai, Tamil Nadu, India, March 2014, No. 3, pp. 1391-1395.
3. *Mudrov V.I.* Zadacha o kommivoyazhere [The problem of the traveling salesman problem]. Moscow: Librokom, 2013, 16 p.
4. Why we are transposing or reversing the directions of all arcs (edges) in the Kosaraju two pass algorithm? Available at: https://www.quora.com/Why-we-are-transposing-or-reversing-the-directionsof-all-arcs-edges-in-the-Kosaraju-two-pass-algorithm (Accessed 02 March 2017).
5. *Kernighan B.W., Lin S.* An efficient heuristic procedure for partitioning graphs. Available at: https://www.cs.princeton.edu/~bwk/btl.mirror/new/partitioning.pdf (Accessed 02 March 2017).
6. *Matthias M., Fikret E.* Genetic Algorithms For Vertex Splitting in DAGs. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.8018&rep=rep1&type=pdf (Accessed 15 March 2017).
7. *Abdel K.I.* Path Partition in Directed Graph-Modeling and Optimization, *New Trends in Mathematical Sciences*. Faculty of Arts and Sciences, Islamic University of Lebanon, 2013, No. 1, pp. 74-84.
8. *Alamdari S., Mehrabian A.* On a DAG Partitioning Problem. Available at: http://www.math.uwaterloo.ca/~amehrabi/Articles/DAG_Partitioning.pdf (Accessed 23 March 2017).
9. *Gladkov L.A., Kureychik V.V., Kureychik V.M.* Geneticheskie algoritmy [Genetic algorithms]. Moscow: Fizmatlit, 2006, 320 p.
10. *Emel'yanov V.V., Kureychik V.M., Kureychik V.V.* Teoriya i praktika evolyutsionnogo modelirovaniya [Theory and practice of evolutionary modeling]. Moscow: Fizmatlit, 2003, 432 p.
11. *Kormen T., Leyzerson Ch., Rivest R., Shtayn K.* Algoritmy: postroenie i analiz [Algorithms: construction and analysis]. Moscow: Vil'yams, 2013, 1328 p.
12. *Kashirina I.L.* Vvedenie v evolyutsionnoe modelirovanie: uchebnoe posobie [Introduction to evolutionary modeling: a textbook], ed. by E.S. Kotlyarovoy. Voronezh: Izd-vo VGU, 2007, 40 p.
13. *Kobak V.G., Kavtaradze I.Sh., Bormotov V.V., Shvidchenko S.A.* Reshenie zadachi kommivoyazhera modifitsirovannoy model'yu Gol'dberga s pomoshch'yu razlichnogo vida mutatsiy [The solution to the traveling salesman problem a modified model of Goldberg with the help of different types of mutations], *Tr. Severo-Kavkazskogo filiala Moskovskogo tekhn. un-ta svyazi i informatiki: mezhdunar. molodezh. nauch.-prakt. konf.* [Proceedings of the North-Caucasian branch of Moscow technical. University of communications and Informatics: international youth scientific and practical conference]. Vol. 1. Rostov-on-Don, 2014, pp. 258-261.
14. *Kobak V.G., Rudova I.Sh.* Vozmozhnosti ispol'zovaniya elitnykh osobey pri reshenii zadachi kommivoyazhera model'yu Gol'dberga [The possibility of the use of elite individuals in solving the traveling salesman problem model Goldberg], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of computer and information technologies], 2016, No. 11 (149), pp. 3-7.
15. *Kobak V.G., Rudova I.Sh., Zhukovskiy A.G.* Sravnitel'nyy analiz modifitsirovannoy modeli Gol'dberga i murav'inogo algoritma pri reshenii zadachi kommivoyazhera [Comparative analysis of a modified model of Goldberg and ant colony optimization algorithm in solving traveling salesman problems], *Tr. Severo-Kavkazskogo filiala Moskovskogo tekhn. un-ta svyazi i informatiki: mezhdunar. molodezh. nauch.-prakt. konf.* [Proceedings Severo-Kavkazskiy branch of Moscow technical University of communications and Informatics: international youth scientific and practical conference]. Vol. 1. Rostov-on-Don, 2015, pp. 362-365.
16. *Kobak V.G., Rudova I.Sh.* Vozmozhnosti ispol'zovaniya elitnykh osobey pri reshenii zadachi kommivoyazhera model'yu Gol'dberga [The possibility of the use of elite individuals in solving the traveling salesman problem model Goldberg], *Vestnik Donskogo gosudarstvennogo tekhnicheskogo universiteta* [Vestnik of DSTU], 2016, Vol. 16, No. 2 (85), pp. 129-135.
17. *Kobak V.G., Rudova I.Sh.* Reshenie zadachi kommivoyazhera modifitsirovannoy model'yu Gol'dberga s ispol'zovaniem razlichnykh sil'nykh mutatsiy [The solution to the traveling salesman problem a modified model of Goldberg, using different strengths of mutation], *Sb. tr. Yubileynoy konf. studentov i molodykh uchenykh, posvyashchennoy 85-letiyu DGTU* [Proceedings of the Anniversary conference of students and young scientists, dedicated to the 85th anniversary of DSTU]. Rostov-on-Don: DGTU, 2015, pp. 146-156.

18. *Kobak V.G., Rudova I.Sh., Zhukovskiy A.G., Shvidchenko S.A.* Ispol'zovanie sil'noy mutatsii pri reshenii zadachi kommivoyazhera [Using strong mutation in solving the traveling salesman problem], *Sovremennye tendentsii razvitiya nauki i tekhnologiy* [Modern trends in the development of science and technology], 2016, No. 6-1.
19. *Rudova I.Sh., Kobak V.G*. Reshenie zadachi kommivoyazhera modifitsirovannoy model'yu Goldenberga s pomoshch'yu sovmestnogo ispol'zovaniya murav'inogo i geneticheskogo algoritmov: svidetel'stvo o gosudarstvennoy registratsii programm dlya EVM. 2016610345; data registratsii 11.01.16 g. [The solution to the traveling salesman problem a modified model of Goldenberg with sharing the ant and genetic algorithms: a certificate of state registration of computer programs. 2016610345; registration date 11.01.16].
20. TSP_LIB. Available at: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/ (Accessed 20 February 17).

Статью рекомендовал к опубликованию д.т.н., профессор В.М. Курейчик.

Кобак Валерий Григорьевич – Донской государственный технический университет; e-mail: valera33305@mail.ru; г. Ростов-на-Дону, пл. Гагарина, 1; тел.: 89185802189; кафедра программного обеспечения вычислительной техники и автоматизированных систем; д.т.н.; профессор.

Рудова Ирма Шалвовна – e-mail: irmuse4ka@rambler.ru; тел.: 89518495586; кафедра программного обеспечения вычислительной техники и автоматизированных систем; аспирант.

Kobak Valerij Grogor'evich – Don State Technical University; e-mail: valera33305@mail.ru; 1, Gagarin sq., Rostov-on-Don, Russia; phone: +79185802189; the department of software computer technology and automated systems; dr. of eng. sc.; professor.

Rudova Irma Shavlovna – e-mail: irmuse4ka@rambler.ru; phone: +79518495586; the department of software computer technology and automated systems; postgraduate student.

УДК 007:519.816 DOI 10.23683/2311-3103-2017-3-148-157

С.М. Ковалев, А.Н. Шабельников

ИЕРАРХИЧЕСКИЙ ИНТЕЛЛЕКТУАЛЬНЫЙ ПРЕПРОЦЕССИНГ НЕЧЕТКО-СТОХАСТИЧЕСКОЙ ИНФОРМАЦИИ В ИНТЕГРИРОВАННЫХ СИСТЕМАХ ДИНАМИЧЕСКОГО ТИПА

Предлагается новая методология обработки первичной информации в интегрированных системах управления динамическими процессами с использованием методологии иерархического интеллектуального препроцессинга. Разрабатываемая методология позволяет не только устранять в первичных данных различного рода шумы и искажения, но также осуществлять их адекватную интерпретацию для последующего принятия решений. В основу иерархического интеллектуального препроцессинга положены модели на основе знаний. На нижних иерархических уровнях ключевую роль играют знания о процессах, порождающих первичную информацию и позволяющие организовать эффективную "очистку" первичных данных от шумов и помех, на верхних уровнях – функциональные знания о технологическом процессе и целях управления, позволяющие повысить эффективность вырабатываемых решений. В качестве базового механизма принятия решений на нижнем уровне интеллектуального препроцессинга выступает гибридная схема нечетко-стохастического вывода, позволяющая одновременно учесть динамичность информации вместе с факторами нечеткостной и стохастической неопределенности. На верхнем уровне полученные результаты отображаются на прагматическую шкалу укрупненных оценок, представленную относительно небольшим числом лингвистических значений, обеспечивающих возможность выработки решений с допустимым уровнем неточности. Принципиальной особенностью предложенной гибридной схемы нечеткого вывода является интеграция в схему вывода стохастическую информацию о законах распределения изме-