ВЕСТНИК ЮЖНОГО НАУЧНОГО ЦЕНТРА РАН, Том 1, № 4, 2005, стр. 87–92

ЗАЩИТА ИНФОРМАЦИИ

УДК 628.3 + 621.3

БИОНИЧЕСКИЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

© 2005 г. В.В. Курейчик[1]

Рассматривается проблема решения задачи о коммивояжере. Для ее решения предлагаются модифицированные методы моделирования эволюции и бионического поиска. Это позволяет получать решение с локальными оптимумами за полиномиальное время. Сложность алгоритмов имеет в среднем квадратичный порядок.

Задача о коммивояжере является одной из ключевых задач искусственного интеллекта. Она нашла широкое применение в инженерных приложениях. Без каких-либо изменений в постановке эта задача используется для проектирования ЭВА и РЭА, разводки коммуникаций, разработки архитектуры вычислительных сетей и др. Кроме того, задача о коммивояжере (ЗК) имеет теоретическую ценность, являясь асимптотической оценкой исследования различных эвристических алгоритмов. Задача коммивояжера (Traveling Salesman problem – TSP) является NP-полной задачей [1–6]. Она заключается в нахождении кратчайшего гамильтонова цикла в графе. Предлагаются бионические подходы для решения этой задачи. В отличие от известных генетических алгоритмов решения данной задачи [2–4, 6] предлагается новая стратегия "эволюция-поиск" "поиск-эволюция" и ее различные иерархические модификации. Разработана программная среда для решения этой задачи. Проведен вычислительный эксперимент и протестирована серия из 1 000 графов на 100, 300, 500, 700 и 1 000 вершин. При этом временная сложность алгоритма не выходила из области полиномиальной сложности. В лучшем случае временная сложность алгоритма ≈ O(nlogn), в худшем случае – $O(n^3)$, где n – число вершин графа.

ПОСТАНОВКА ЗАДАЧИ

Постановка задачи записывается следующим образом: коммивояжеру необходимо посетить W городов, не заезжая в один и тот же город дважды, и вернуться в исходный пункт по маршруту с минимальной стоимостью. Имеем: дан полный граф $G = (X, U)$, где $|X| = n$ – множество вершин (города), $|U| = m$ – множество ребер (возможные пути между городами). Дана матрица смежности $R(i, j)$, где $i, j \in 1,2,..., n$, – задающая стоимости путей из вершины x_i в x_j. Требуется найти перестановку из элементов множества X такую, что целевая функция (ЦФ) [1, 2, 5, 8]

$$\text{ЦФ}(\varphi) \equiv R(\varphi(1), \varphi(n)) + \\ + \Sigma\{R(\varphi(i),\varphi(i+1))\} \rightarrow \min. \quad (1)$$

Целевой функцией является суммарная стоимость ребер гамильтонова цикла заданного графа. Решение ЗК будем рассматривать для графов с весами на ребрах. Отметим, что значение ЦФ не зависит в частном случае от выбора вершины начала маршрута. Фиксация вершины-старта может быть использована для сужения пространства поиска, которое в этом случае равно $(n – 1)!$ Поэтому в общем случае результаты решения ЗК чувствительны к выбору начальных условий. Время работы известных алгоритмов ЗК, основанных на полном переборе или на построении дерева решений (методы ветвей и границ и др. [1, 7–10]), растет экспоненциально в зависимости от числа вершин графа. Эти алгоритмы способны решать ЗК для малого числа вершин ($n \approx 50$) за полиномиальное время. Поэтому для решения ЗК с $n > 50$ используются различные эвристики.

Приведем ряд эвристик для эффективного решения ЗК [7–10].

Эвристика 1 – это предпочтение более невыгодного маршрута более выгодному с определенной вероятностью.

Эвристика 2 – анализ 50% случайных маршрутов и 50% маршрутов, использующих знания о решаемой задаче.

Эвристика 3 – если путь оказывается тупиковым, производится случайный выбор вершины

[1] Таганрогский государственный радиотехнический университет, г. Таганрог.

или же выбор ближайшей вершины из числа не рассмотренных для включения в ЗК.

Эвристика 4 – Получение различных наборов альтернативных решений и применение бионических методов для повышения качества решений.

ОПИСАНИЕ АРХИТЕКТУРЫ ПОИСКА И ОСНОВНЫХ БЛОКОВ АЛГОРИТМА

Структурная схема бионического алгоритма решения ЗК на основе моделей эволюций Ч. Дарвина, Ж. Ламарка, де Фриза, К. Поппера и М. Кимуры [6, 10] приведена на рисунке 1. В алгоритме начальная популяция конструируется тремя способами: A_1, A_2, A_3. В случае A_1 популяция решений (заданных путей) формируется случайным образом. В случае A_2 популяция решений получается путем неоднократного применения последовательного алгоритма. Формирование популяции A_3 производится совместно случайным и направленным образом. Отметим, что лицо, принимающее решение (ЛПР) на основе блока эволюционной адаптации, может создавать любое число популяций другими способами. В архитектуру (рис. 1) введен блок локального поиска, который позволяет получать локальные экстремумы в ЗК. Все генетические операторы ориентированы на использовании знаний о решаемой задаче. Блок редукции позволяет оставлять размер популяции постоянным для каждой генерации. После формирования популяции к каждому ее элементу применяются различные генетические операторы кроссинговера (ОК), инверсии, мутации (ОМ), сегрегации, транслокации. В данной схеме порядок выполнения генетических операторов зависит от внешней среды, блока эволюционной адаптации и может иметь любой установленный порядок. Генетический оператор по аналогии с оператором алгоритма – это средство отображения одного множества на другое. Другими словами, это конструкция, представляющая один шаг из последовательности действий генетического алгоритма. Это языковая конструкция, позволяющая на основе различных преобразований альтернативных решений (родителей) или их частей создавать новые решения (потомки). Блок эволюционной адаптации задает хаотичный (случайный) порядок выполнения генетических операторов. Заметим, что перед началом работы алгоритма предварительно вычисляется верхняя оценка минимального пути. После выполнения генетических операторов их результаты сравниваются с оценкой. При получении искомых результатов алгоритм заканчивает свою работу. Затем формируется новая популяция, и процесс продолжается аналогично далее. Возможен выход из каждого оператора в отдельности при получении локального оптимума. Блок редукции варьирует размер популяции. При попадании в локальный оптимум предлагается выполнять следующие эвристики.

– Изменять глубину "горизонта".

– Постоянство одной хромосомы: из всех маршрутов, имеющих одинаковую длину, только один остается неизменным, над всеми другими выполняются модифицированные генетические операторы.

– Использовать различные обратные связи для изменения параметров генетического поиска.

Отметим, что такие генетические операторы и эвристики работают не всё время, а только в ситуациях, когда однообразие в популяции достигает высокого уровня. Совместная эволюция Ч. Дарвина, Ж. Ламарка, де Фриза и К. Поппера регламентирует и управляет процессом генетического поиска.

В основном цикле бионического алгоритма промежуточная популяция решений не создавалась, получаемый потомок замещал наихудшее решение (хромосому) в популяции. Двумя основными параметрами при решении задачи о коммивояжере, оценивающими эвристический алгоритм, являются скорость сходимости и близость получаемого решения к оптимальному (если оно известно) или же к лучшим стандартным решениям (бенчмаркам), полученным с помощью других эвристик (если оптимум не известен).

Существуют два основных генетических представления возможного решения: кодирование решения в виде списка вершин (последовательность посещаемых городов) или кодирование решения как списка ребер, образующих гамильтонов цикл (список использованных путей из города в город) [2–6]. Кроме этого интерес представляет матричное кодирование задачи о коммивояжере. В матрице единицами кодируются переходы между генами в хромосоме. Тогда популяция хромосом будет состоять из набора аналогичных матриц. Такое кодирование ввиду большого числа нулей избыточное, хотя является наглядным и удобным для преобразований. Предложим вместо матриц использовать списки связности. Это позволит получить набор простых строительных блоков. На данном наборе на основе оператора сегрегации и оператора инверсии можно эффективно получать новых допустимых потомков.

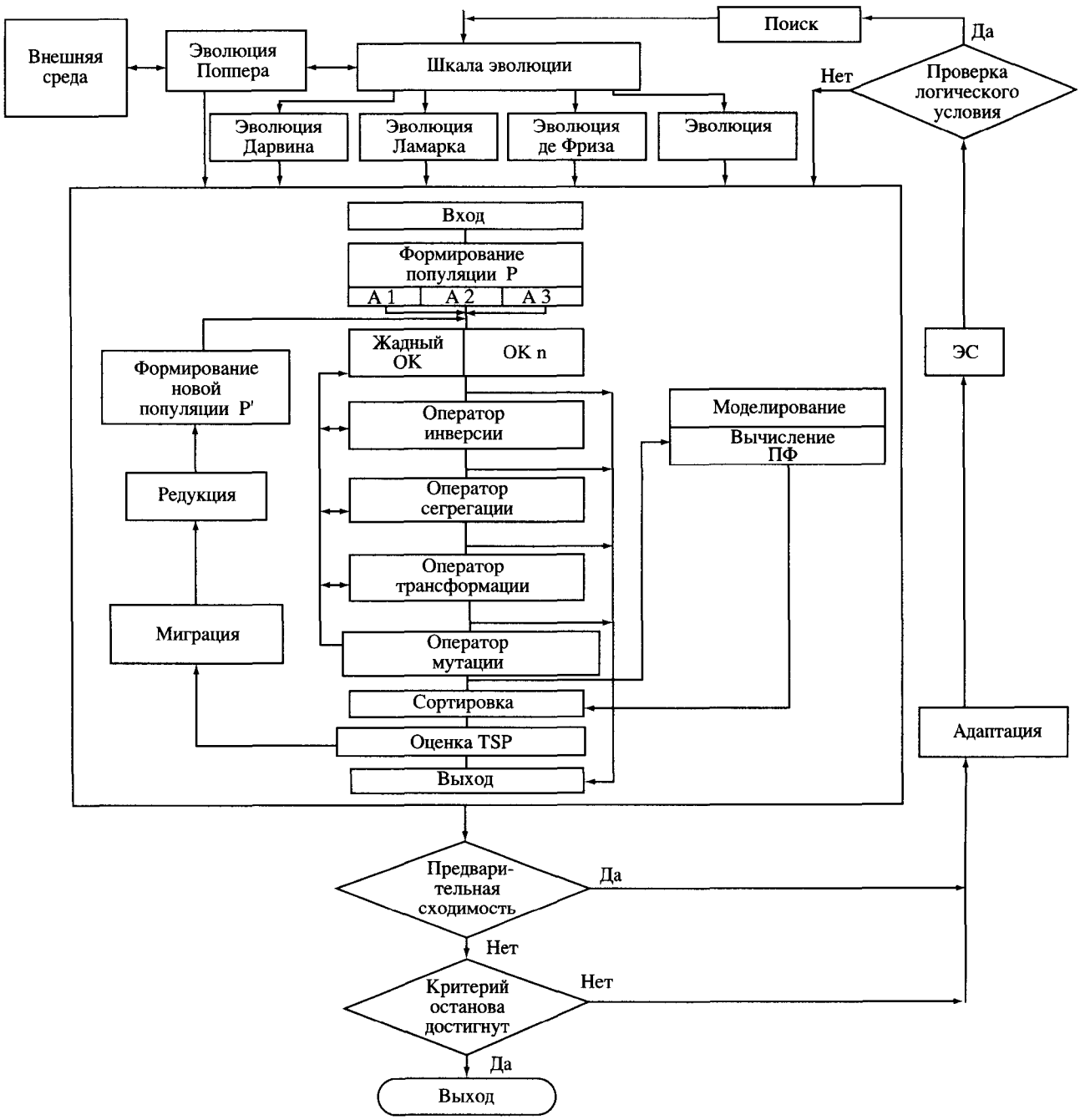

Рис. 1. Структурная схема алгоритма решения задачи о коммивояжере

Для задачи о коммивояжере существуют специально разработанные генетические операторы (ГО), позволяющие гарантированно получать из родителей допустимых потомков [2–4, 6]. Опишем модифицированный комбинированный ОК.

1. Случайным образом выбрать стартовую вершину в первом родителе.

2. Текущий родитель – первый родитель.

3. Тело комбинированного ОК определить так. Выбрать, какая из соседних вершин (соседними являются вершины, расположенные в хромосоме справа и слева от рассматриваемой) в анализируемом родителе ближе к текущей вершине. Ближайшая из соседних вершин, не вошедшая в путь задачи о коммивояжере, становится новой текущей вершиной. Если только одна из двух соседних вершин не вошла в путь задачи о коммивояжере, то она становится текущей. Если обе соседние вершины являются посещен-

ными (тупиковая ситуация), то выбрать ближайшую из числа еще не посещенных.

4. Рассмотреть несколько ветвей дерева решений на основе заданной глубины поиска.

5. Повторять тело комбинированного ОК, пока все вершины не будут посещены. Потомок формируется как последовательность посещаемых вершин.

6. Конец работы алгоритма.

Опишем построение генетических операторов с использованием метода дихотомии. Они реализуются за счет механизма перебора точек разрыва [6]. Приведем нечеткий алгоритм построения ОК дихотомии (ОКД).

1. Пусть заданы две родительские хромосомы длины L.

2. Делим хромосому (отрезок L) пополам (при нечетном размере хромосомы в любую часть берется большее ближайшее число генов).

3. Точка разрыва определяет точку ОК дихотомии.

4. По правилам построения одноточечного ОК получаем две новых хромосомы-потомка.

5. Каждую половину хромосомы-потомка снова делим на две части. Процесс продолжаем аналогично до тех пор, пока не будет получено заданное количество хромосом-потомков или алгоритм по методу дихотомии завершится. При получении нереальных решений (хромосом с повторяющимися генами) производится восстановление реальных решений (повторяющие гены меняются на отсутствующие гены из хромосом-родителей).

6. Конец работы алгоритма.

Вероятность выживания альтернативного решения с лучшим значением ЦФ после первого шага оператора вычисляется по формуле:

$$P_r(s)[\text{ОКД}] = 1 - 2^0 / L - 1. \quad (2)$$

Вероятность выживания альтернативного решения с лучшим значением ЦФ после второго и последующих шагов оператора определится по формуле:

$$P_r(s)[\text{ОКД}] = 1 - 2^1 / L - 1. \quad (3)$$

Эффективность метода дихотомии экспоненциально растет с ростом числа хромосом. Отметим, что существует большое количество способов конкретной реализации ОК-дихотомии. Их целесообразность определяется на основе вероятности выживания лучших хромосом (альтернативных решений).

Построение ОМ Фибоначчи (ОМФ) реализуется за счет аналогичного методу дихотомии механизма перебора точек разрыва. Приведем нечеткий алгоритм построения ОМ с использованием метода Фибоначчи, ориентированный на ЗК. В заданной популяции хромосом на основе оператора репродукции выбирается родительская хромосома (альтернативное решение) длины L с наименьшим значением ЦФ [6].

1. В выбранной хромосоме определяем точку разрыва. Она соответствует третьему числу ряда Фибоначчи.

2. Производим реализацию стандартного ОМ и получаем новую хромосому-потомка.

3. Вычисляем значение ЦФ хромосомы-потомка. Если получено решение, удовлетворяющее заданным конструкторским ограничениям на компоновку, то конец работы алгоритма.

4. Далее в качестве точки ОМ Фибоначчи выбираем 4, 5,... числа ряда Фибоначчи и переходим к пункту 3. Алгоритм оканчивает работу по достижению заданного критерия или когда номер числа ряда Фибоначчи $\geq L$.

5. Конец работы алгоритма.

Данный генетический оператор эффективно применяется при компоновке блоков, когда число элементов в блоках равное или близкое к числам Фибоначчи. Целесообразность ОМ Фибоначчи определяется на основе знания о решаемой задаче компоновки и вероятности выживания лучших решений после его применения. Вероятность выживания альтернативного решения с лучшим значением ЦФ после реализации оператора вычисляется по формуле:

$$P_r(s)[\text{ОМФ}] = 1 - 1 / L - 1. \quad (4)$$

Отметим, что ОК Фибоначчи практически совпадает с многоточечным ОК. Отличие заключается в том, что точки разреза в ОК Фибоначчи являются фиксированными и соответствуют числам Фибоначчи.

Приведем алгоритм построения модифицированного оператора транслокации (ОТ) для ЗК на основе жадной стратегии.

1. Для каждой пары хромосом случайным образом выберем точку жадного ОТ и в качестве номера стартовой вершины графа возьмем номер отмеченного гена в хромосоме.

2. Сравним частичную стоимость путей, ведущих из текущих вершин в анализируемые, и выберем из них путь с наименьшей стоимостью.

3. Если выбранная таким образом вершина графа уже была включена в частичный путь, то взять случайную вершину из числа еще не отмеченных. Присвоить полученной таким образом вершине значение текущей.

4. При преждевременном образовании циклов выбирается другой кратчайший путь.

5. Пробное инвертирование элементов всего пути или его частей с выбором наилучшего.

6. Повторяем пункты 2 и 3, пока не будет построен гамильтонов цикл с минимальной суммарной стоимостью ребер.

7. Конец работы алгоритма.

Жадные ОТ, основанные на знаниях о задаче, улучшают скорость сходимости алгоритма. Когда путь оказывается тупиковым, предлагается комбинация жадной стратегии и метода "горизонта".

РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНОГО ЭКСПЕРИМЕНТА

Рассмотрим кратко результаты экспериментальных исследований задачи определения пути коммивояжера. Интерфейс программы рассчитан на функционирование в среде Windows 95/98/2000/NT. Он реализован на языке Си++ в Borland Builder 5.0.

В программе в разделе "Статистика" приводятся начальная и конечная популяции с их значениями целевой функции, а также гистограмма средней приспособленности популяций. Дается суммарная величина целевой функции начальной и конечной популяции. Временная сложность представленного алгоритма рассчитывается как время, затраченное в среднем на одну итерацию алгоритма. Эта зависимость приведена на рисунке 2. Здесь величина I обозначает количество итераций, при котором производилось исследование, а n – размер популяции.

При исследовании стандартного теста Eilon's-75 результаты совпали с наилучшими существующими [3, 4]. Лучший путь для Eilon's-75 составлял 545 условных единиц [3]. Описанный алгоритм определил конфигурации наилучшего маршрута. Новое лучшее решение в Eilon's-75 составляет 535, при этом размер популяции равен 20, уровень мутаций равен 0.3, а элитное число равно 10. Приведем координаты этого маршрута: вершина 29 с координатами (30,20), 28 (27,24), 27 (22,22), 26 (26,29), 25 (20,30), 24 (21,36), 47 (21,45), 23 (21,48), 22 (22,53), 21 (26,59), 45 (30,60), 46 (35,60), 49 (40,60), 19 (35,51), 20 (30,50), 18 (33,44), 17 (29,39), 16 (33,34), 15 (38,33), 14 (40,37), 13 (45,35), 12 (45,42), 11 (41,46), 10 (50,50), 9 (55,50), 57 (55,57), 56 (62,57), 55 (70,64), 54 (57,72), 53 (55,65), 52 (50,70), 48 (47,66), 50 (40,66), 51 (31,76), 43 (10,70), 44 (17,64), 42 (15,56), 41 (9,56), 40 (7,43), 39 (12,38), 38 (11,28), 37 (6,25), 36 (12,17), 35 (16,19), 34 (15,14), 33 (15,5), 32 (26,13), 31 (36,6), 72 (44,13), 71 (50,15), 70 (54,10), 69 (50,4), 68 (59,5), 67 (64,4), 66 (66,8), 65 (66,14), 64 (60,15), 63 (55,20), 62 (62,24), 61 (65,27), 60 (62,35), 59 (67,41), 58 (62,48), 8 (55,45), 7 (51,42), 6 (50,40), 5 (54,38), 4 (55,34), 3 (50,30), 2 (52,26), 1 (48,21), 75 (43,26), 74 (36,26), 73 (40,20), 30 (35,16).

Далее была протестирована серия из 1 000 графов на 100, 300, 500, 700 и 1 000 вершин. При этом временная сложность алгоритма не выходила из области полиномиальной сложности. В лучшем случае временная сложность алгоритма ≈ ≈ O(nlogn), в худшем случае – $O(n^3)$. Отметим, что отличительной особенностью разработанного алгоритма является способность хорошо работать на популяциях с малым числом хромосом, что уменьшает время реализации алгоритма.

В заключение отметим, что временная сложность описанных алгоритмов является полиномиальной величиной и лежит в пределах от $O(n)$ до $O(n^3)$.

Работа выполнена при поддержке РФФИ, грант 03-01-00336.

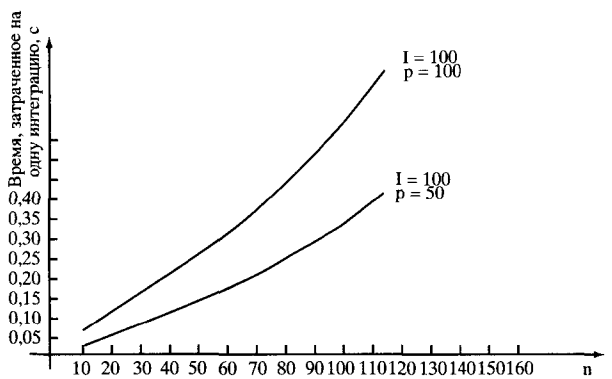

Рис. 2. График зависимости времени решения от числа вершин графа

ЛИТЕРАТУРА

1. *Кормен Т., Лейзерсон И., Ривест Р.* Алгоритмы: построения и анализ. М.: Изд-во МЦНМО, 2000. 960 с.
2. *Емельянов В.В., Курейчик В.В., Курейчик В.М.* Теория и практика эволюционного моделирования. М.: Физматлит, 2003. 432 с.
3. *Goldberg David E.* Genetic Algorithms in Search, Optimization and Machine Learning. USA: Addison-Wesley Publishing Company, Inc., 1989. 412 p.
4. Handbook of Genetic Algorithms / Edited by Lawrence Davis. USA: New York, Van Nostrand Reinhold, 1991. 385 p.
5. *Grefenstette J., Gopal G., Rosmaita B.* // Proc. Intern. Conf. of Genetic Algorithms and their applications. John Grefenstette. New Jersey, 1987. P. 160–165.
6. *Oliver I., Smith D., and Holland J.R.* // Proc. of the Second International Conf. on Genetic Algorithms, 1987. P. 224–230.

7.*Kureichik V.M., Miagkikh V.* // Proc. of the second International conference. Plymouth UK, 1996. P. 294–296.
8.Genetic Algorithms and Evolution Strategy in Engineering and Computer Science / Eds D. Quagliarella and et all. John Wiley & Song, NY, USA, 1998. P. 95–103.
9.*Potts C.I., Giddens T.D., Yadav S.B.* // IEEE Trans. on Systems, Man and Cybernetics, 1994. V. 24. № 1. P. 73–86.
10. *Tsuya A., Tanaka A.* // Proc. 2001 Int. symposium on nonlinear theory and its applications. Miyagi, Japan, 2001. P. 319–322.

DECISIONS OF TRAVELING SALESMAN PROBLEM BY BIONIC METHODS

© 2005 V.V. Kureichik

The Traveling Salesman problem is considered. The modified methods of evolution simulation and bionic search are offered for its decision. It allows receiving the decision with local optimum for polynomial time. Complexity of algorithms has a square-law order on the average.

REFERENCES

1. Kormen T., Leyzerson I., Rivest R. 2000. *Algoritmy: postroeniya i analiz*. [*Algorithms. Construction and analysis*]. Moscow: 960 p. (In Russian).
2. Emel'yanov V.V., Kureychik V.V., Kureychik V.M. 2003. *Teoriya i praktika evolyutsionnogo modelirovaniya*. [*Theory and practice of evolutionary modeling*]. Moscow, Fizmatlit Publ.: 432 p. (In Russian).
3. Goldberg David E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. USA: Addison-Wesley Publishing Company, Inc.: 412 p.
4. Handbook of Genetic Algorithms. 1991. (Edited by Lawrence Davis). USA, New York, Van Nostrand Reinhold: 385 p.
5. Grefenstette J., Gopal G., Rosmaita B. 1987. In: *Proc. Intern. Conf. of Genetic Algorithms and their applications*. New Jersey: 160–165.
6. Oliver I., Smith D., and Holland J.R. 1987. In: *Proc. of the Second International Conf. on Genetic Algorithms.* P. 224–230.
7. Kureichik V.M., Miagkikh V. 1996. In: *Proc. of the second International conference*. Plymouth UK: 294–296.
8. *Genetic Algorithms and Evolution Strategy in Engineering and Computer Science*. 1998. (Eds D. Quagliarella and et all.). John Wiley & Song, NY, USA: 95–103.
9. Potts C.I., Giddens T.D., Yadav S.B. 1994. *IEEE Trans. on Systems, Man and Cybernetics*. 24(1): 73–86.
10. Tsuya A., Tanaka A. 2001. *Proc. 2001 Int. symposium on nonlinear theory and its applications, Miyagi, Japan*. P. 319–322.