Весці БДПУ. Серыя 3. 2019. № 3. С. 94–101.

УДК 004.421.2

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ОПЕРАТОРОВ КРОССОВЕРА PMX, CX И OX НА ПРИМЕРЕ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

С. М. Гардейчик,
аспирант Белорусского государственного педагогического университета имени Максима Танка

Поступила в редакцию 5.07.2019.

UDC 004.421.2

COMPARATIVE ANALYSIS OF OPERATORS OF THE CROSSOVER PMX, CX AND OX ON THE EXAMPLE OF SOLVING THE PROBLEM OF THE TRAVELING SALESMAN

S. Gardeychik,
Postgraduate Student Belarusian State Pedagogical University named after Maxim Tank

Received on 5.07.2019.

Задача коммивояжера (Travelling Salesman Problem, TSP) – одна из самых известных NP-трудных задач комбинаторной оптимизации, заключающаяся в поиске самого выгодного маршрута, проходящего через все указанные города ровно один раз с последующим возвратом в исходный город. Генетический алгоритм (ГА) является одним из лучших методов, который используется для решения различных NP-трудных задач, таких, как TSP. Стандартные операторы кроссовера (1-Point Crossover, k-Point Crossover и др), порождают потомков, которые в общем случае не являются перестановками. Цель статьи заключается в отыскании наилучшего возможного результата кроссинговера комбинаторной задачи оптимизации, при заданных двух родительских решениях, закодированных в виде перестановок. В работе проводится экспериментальное исследование операторов оптимальной рекомбинации PMX, CX и OX для задачи TSP. Экспериментальные результаты показывают превосходство оператора кроссовера PMX, поскольку он находит наиболее оптимальное решение.

Ключевые слова: TSP, NP-трудность, генетический алгоритм, мутация, кроссовер, селекция, PMX, CX, OX.

The Traveling Salesman Problem (TSP) is one of the most well-known NP-hard combinatorial optimization problems, which consists in finding the most profitable route that passes through all the specified cities exactly once and then returns to the original city. The genetic algorithm (GA) is one of the best methods used to solve various NP-hard problems, such as TSP. Standard crossover operators (1-Point Crossover, k-Point Crossover, etc.) generate descendants that are not generally permutations. The purpose of the paper is to find the best possible result of a cross-over combinatorial optimization problem, with the given two parental solutions coded as permutations. An experimental study of the optimal recombination operators PMX, CX and OX for the TSP problem is carried out. The experimental results show the superiority of the PMX crossover operator, since it finds the most optimal solution.

Keywords: TSP, NP-hard, Genetic Algorithm, Mutation, Crossover, Selection, PMX, CX, OX

Введение. В классическом подходе хромосома, которая кодирует решение (путь коммивояжера), состоит из N генов (количество городов). Каждый ген содержит номер, который является меткой города. Таким образом, хромосома является прямым кодированием перестановки – упорядоченного набора 1, 2, ..., *n* без повторения чисел.

Так как популяция состоит из «действительных» хромосом, стандартные операторы кроссовера и мутации вызовут ряд проблем. Потомство, полученное с помощью обычных операторов кроссовера, имеет высокую вероятность быть недействительным, если одни метки городов пропали, а другие повторились.

Исправить положения с недействительными решениями можно тремя способами:

1. Дисквалификация. Идея состоит в том, чтобы позволить генерировать недействительные хромосомы, с назначением низких значений пригодности, такие хромосомы будут устранены в предстоящем процессе отбора. Недостатком этого простого метода является длительность. Генетический алгоритм (ГА) большую часть времени выполнения генерирует заведомо неправильные хромосомы и затем устраняет их.

2. Восстановление. При таком подходе генерируются недействительные хромосомы, затем запускается промежуточный процесс, где они преобразуются в действительные. Здесь ключевая идея состоит в том, чтобы задействовать минимальное количество модификаций, чтобы преимущества кроссовера были сохранены. Данный способ также, как и предыдущий, требует временных затрат.

3. Разработка специализированных операторов. Вместо генерации недопустимых хромосом операторы кроссовера ГА модифицируются, чтобы порождать только допустимые хромосомы.

К третьей категории относятся операторы скрещивания, в которых учтены ограничения, накладываемые на генерацию перестановок: Partially – Mapped Crossover (PMX), Order – Crossover (OX1), Order Based Crossover (OX2), Position Based Crossover (POS), Heuristic Crossover (HX), Edge Recombination Crossover (ER), Sorted Match Crossover (SMX), Maximal Preservative Crossover (MPX), Voting Recombination Crossover (VR), Alternating – Position Crossover (AP) и др.

Генетический алгоритм с операторами PMX, CX и OX. Был предложен Джоном Холландом в 1975 г. Является разновидностью эволюционных вычислений, с помощью которых решаются оптимизационные задачи с использованием методов естественной эволюции, таких, как наследование, мутации, селекция (отбор) и кроссинговер (кроссовер, скрещивание). Отличительной особенностью генетического алгоритма является акцент на использование оператора «скрещивания», который производит операцию рекомбинации решений-кандидатов, роль которой аналогична роли скрещивания в живой природе. Основные шаги ГА решения задачи TSP приведены ниже:

(A) Схема кодирования

Каждый город ассоциируется с числовой меткой, например, если есть 9 городов, то каждому городу присваивается метка от 1 до 9 (все числовые метки попарно различны). Запись 9→1 →8 → 2 → 4 → 3 → 6 → 7 → 5 описывает направление движения коммивояжера. Для *N* числа городов схема кодирования хромосомы представляет собой перестановку целых чисел от 1 до *N*.

(B) Создание начальной популяции

Начальная популяция случайным образом генерируется из множества, которое содержит все возможные решения, после чего происходит ее улучшение через итерационное применение операторов отбора и рекомбинации. В случае задачи TSP на N городов пространство поиска состоит из N! вариантов, в котором ищется оптимальное решение.

(C) Функция приспособленности (ФП)

ФП позволяют определить, какие особи будут выбраны для рекомбинации и мутации на каждой фазе ГА. Для задачи TSP значение пригодности присваивается каждому решению (хромосоме) путем расчета расстояния между городами (генами) в каждой хромосоме. Например, для хромосомы вида $I_i : a_1, a_2, ..., a_n$, где:

1) I_i – *i*-хромосома в популяции и $i = \overline{1, Population\,Size}$;

2) *N* – количество городов (генов) в решении (хромосоме).

ФП *i*-хромосомы вычисляется по формуле:

$$fitness_i = dist(a_1, a_N) + \sum_{i=1}^{N-1} dist(a_i, a_{i+1}),$$

где $dist(a_i, a_{i+1})$ – евклидово расстояние между двумя городами.

(D) Селекция (Отбор)

В каждом последующем поколении для выведения нового поколения отбирается определенная доля особей. Выборка производится с помощью стохастического процесса, основанного на значении ФП. Как правило, особи с наибольшим значением пригодности с большей вероятностью будут отобраны в новое поколение.

При турнирной селекции все особи популяции разбиваются на подгруппы с последующим выбором в каждой из них особи с наилучшей приспособленностью. Подгруппы могут иметь произвольный размер, но чаще всего популяция разделяется на подгруппы по 2–3 особи в каждой.

Турнирный метод пригоден для решения задач как максимизации, так и минимизации функции. В турнирном методе допускается изменение размера подгрупп, на которые подразделяется популяция (tournament size). Исследования подтверждают, что турнирный метод действует эффективнее, чем метод рулетки.

На рисунке 1 представлена схема, которая иллюстрирует метод турнирной селекции для подгрупп, состоящих из двух особей. Такую схему легко обобщить на подгруппы большего размера.

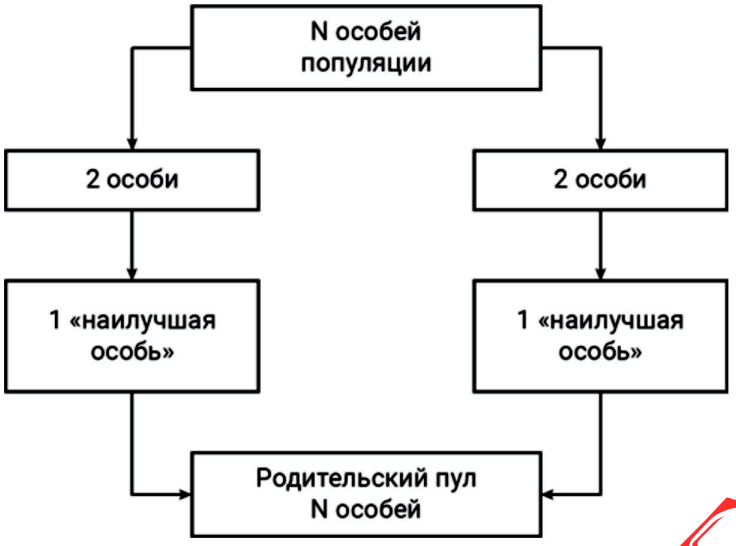

Рисунок 1 – Схема турнирной селекции для подгрупп, состоящих из двух особей

(E) Кроссовер (Скрещивание)

Partially – Mapped Crossover. Был предложен Голдбергом и Лингле. После равномерного выбора двух произвольных точек разреза родительских хромосом, части между точками разреза одного родителя отображаются на строку другого родителя, а оставшиеся гены обмениваются. Рассмотрим, например, две родительские хромосомы (точки разреза обозначим как «|»):

Родитель 1: 1 2 3 | 4 5 6 | 7 8

Родитель 2: 3 7 5 | 1 6 8 | 2 4

Отображение: 4 ↔1, 5 ↔ 6 , и 6 ↔ 8.

Далее, подстрока 1-го родителя копируется 2-му потомку (с соблюдением позиций генов), а подстрока 2-го родителя копируется 1-му потомку.

Потомок 1: *** | 1 6 8 | **

Потомок 2: *** | 4 5 6 | **

Потомок *i*, где *i* = 1, 2 заполняется путем копирования генов *i*-го родителя. В случае, если город уже присутствует в *i*-м потомке, он заменяется в соответствии с отображением. Например, первый элемент 1-го потомка будет равен 1, как и 1-й элемент первого родителя. Однако элемент, значение которого равно 1, уже содержится в потомке. Следовательно, применяя отображение (1 → 4) устанавливаем 1-м элементом 1-го потомка ген, равный 4. Второй, третий и седьмой элементы 1-го потомка можно перенять от первого родителя. Однако последним элементом 1-го потомка будет 8, которое уже присутствует в нем. Применив отображение (8 → 6 → 5), получим значение, равное 5 (рисунок 2а).

Потомок 1: 4 2 3 | 1 6 8 | 7 5

Для формирования 2-го потомка необходимо проделать аналогичные действия (см. рисунок 2б).

Потомок 2: 3 7 8 | 4 5 6 | 2 4

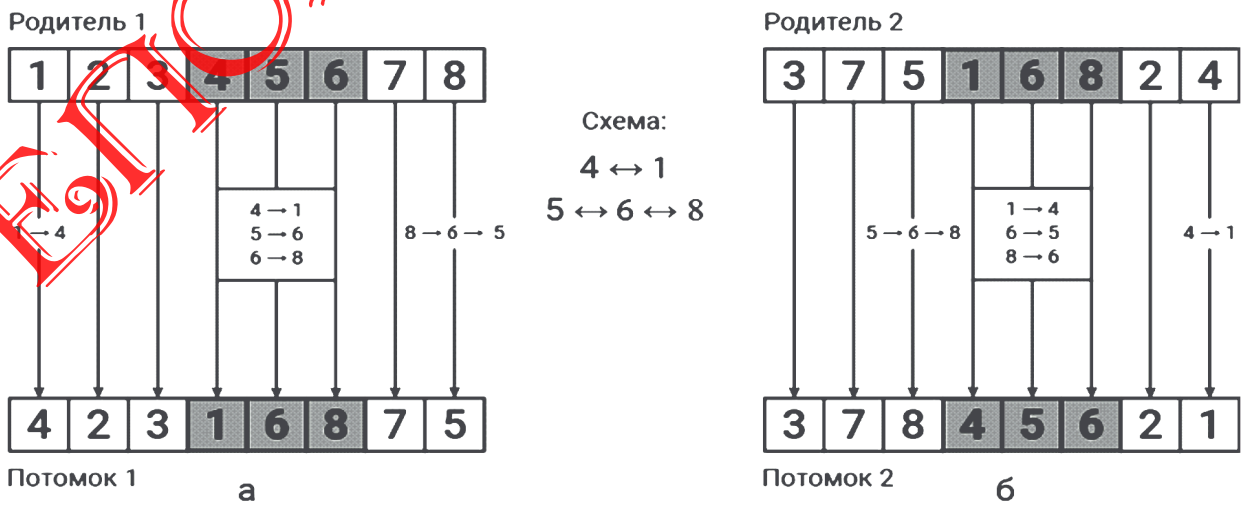

Рисунок 2 – Partially – Mapped Crossover

Cyclic Crossover. Оператор определяет ряд так называемых циклов между двумя родительскими хромосомами. Затем, чтобы сформировать первого потомка, значения первого цикла копируется от первого родителя, значения второго цикла от второго родителя, значения третьего цикла от первого родителя и так далее. Пример:

Родитель 1: 1 2 3 4 5 6 7 8

Родитель 2: 2 4 6 8 7 5 3 1

Цикл 1: Начнем с первого значения у 1-го родителя и опустимся на ту же позицию у 2-го родителя: 1 → 2. Затем ищем 2 у 1-го родителя и находим его на 2-й позиции, где опускаемся до 4: 1 → 2 → 4. Опять же, мы ищем это значение у 1-го родителя и находим его на 4-й позиции и опускаемся до 8: 1 → 2 → 4 → 8. Ищем 8 у 1-го родителя и находим ее на 8-й позиции и опускаемся до 1: 1 → 2 → 4 → 8 → 1. Цикл завершен (рисунок 3а).

Значения 1-го цикла: 1 2 4 8

Родитель 1: 1 2 **3** 4 **5 6 7** 8

Родитель 2: 2 4 **6** 8 **7 5 3** 1

Цикл 2: 3 → 6 → 5→ 7 → 3 (см. рисунок 3б).

Значения 2-го цикла: 3 5 6 7

Родитель 1: 1 2 **3** 4 **5 6 7** 8

Родитель 2: 2 4 **6** 8 **7 5 3** 1

Наполнение потомков значениями.

Копирование 1-го цикла: Значения 1-го цикла из 1-го родителя копируются 1-му потомку, значения от 2-го родителя будут скопированы 2-му потомку.

Копирование 2-го цикла: Значения 2-го цикла из 1-го родителя копируются 2-му потомку, значения от 2-го родителя будут скопированы 1-му потомку.

Сформированы два потомка (рисунок 3в):

Потомок 1: 1 2 6 4 7 5 3 8

Потомок 2: 2 4 3 8 5 6 7 1

Order Crossover. Был предложен Дэвисом в 1985 г. Оператор формирует потомка, выбирая подстроку одного из родителей и сохраняя относительный порядок генов другого родителя. Рассмотрим, например, двух родителей (со случайными двумя точками разреза, отмеченными «|»):

Родитель 1: 3 4 8 | 2 7 1 | 6 5

Родитель 2: 4 2 5 | 1 6 8 | 3 7

Изначально подстрока первого родителя копируется второму потомку, а подстрока второго родителя – первому потомку:

Потомок 1: *** | 1 6 8 | **

Потомок 2: *** | 2 7 1 | **

Далее, в первый потомок, начиная со второй точки разрыва, копируются города из первого родителя, пропуская значения, уже присутствующие в построенной подстроке потомка. По достижении конца списка этот процесс продолжается с первой позиции и до первой точки разрыва (по кольцу).

После второй точки разрыва в первом родителе получаем следующую последовательность: 6 → 5→ 3→ 4 → 8 → 2 →7 → 1. Удаляем из нее города 1, 6, 8, поскольку они уже есть в первом потомке, и в результате полу-

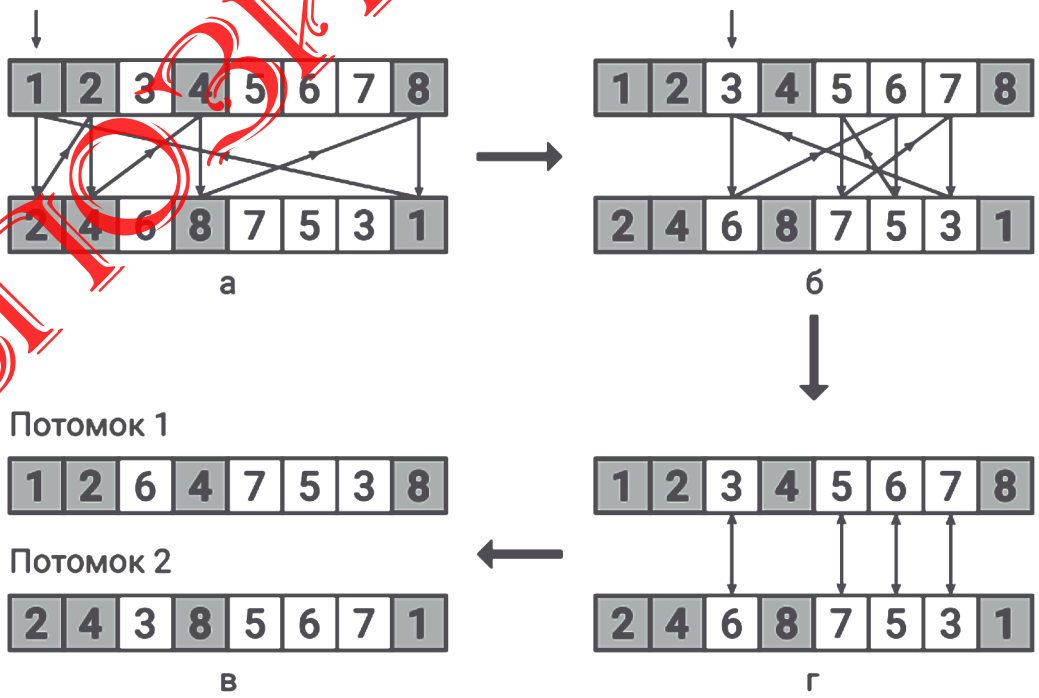

Рисунок 3 – Cycle Crossover

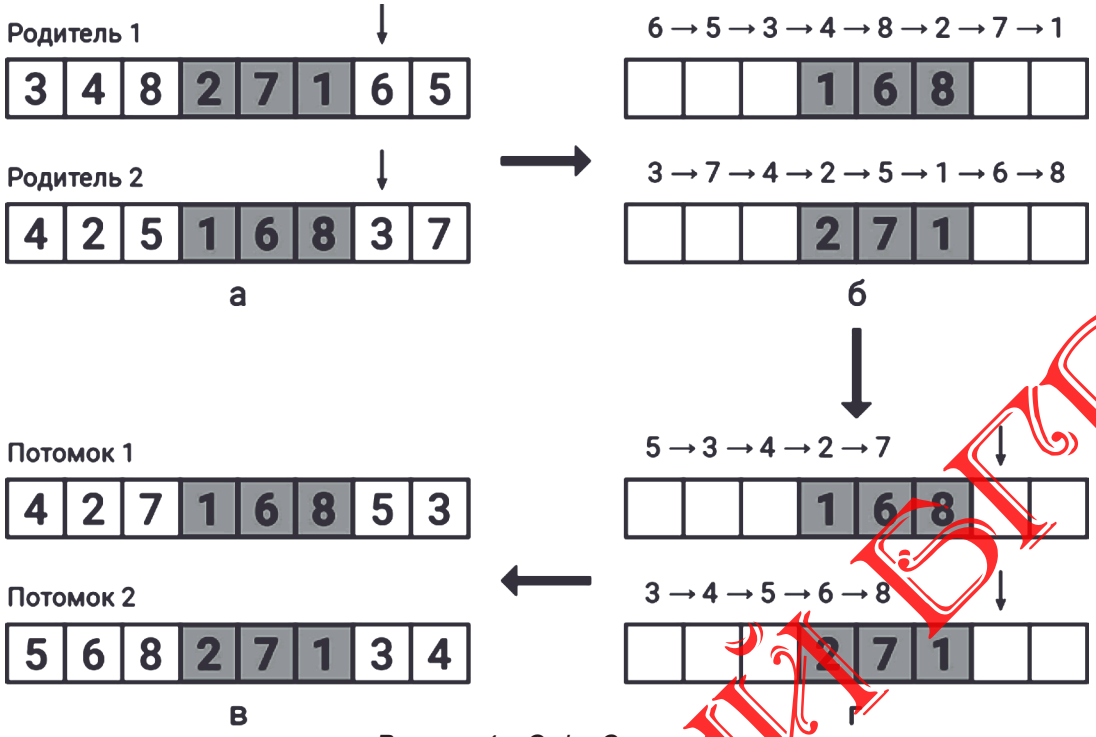

Рисунок 4 – Order Crossover

чаем последовательность 5→ 3→ 4 → 2 → 7. Ее мы помещаем в первый потомок, начиная со второй точки разрыва (по кольцу) и получаем первого потомка.

Потомок 1: 4 2 7 | 1 6 8 | 5 3

Аналогично получаем второго потомка (рисунок 3).

Потомок 2: 5 6 8 | 2 7 1 | 3 4

(F) Мутация

После операции скрещивания хромосомы подвергаются мутационному процессу (mutation). Данный оператор необходим для «выбивания» популяции из локального экстремума и препятствует преждевременной сходимости. Это достигается за счет того, что изменяется случайно выбранный ген (гены) в хромосоме.

При решении текущей задачи мутация производилась путем инвертирования порядка генов между двумя случайно выбранными точками разрыва.

Например: 1 2 3 4 5 6 7 8

Случайным образом выбрана подстрока, результатом будет:

1 2 | 3 4 5 | 6 7 8 → 1 2 | 5 4 3 | 6 7 8

Вероятность мутации принимается равной 0,1. Если установлено высокое значение вероятности, будет наблюдаться примитивный случайный поиск.

(G) Завершение работы алгоритма

Когда алгоритм выполнит заданное число итераций (1000, 2000 и т. д.), он останавливается и выводит наилучшее решение.

Результаты. Решение задачи TSP, с использованием операторов PMX, CX и OX реализовано в пакете MatLab. В качестве испытательного стенда выбраны задачи из библиотеки TSPLIB [3]. Библиотека предоставляется и поддерживается Исследовательской группой по дискретной оптимизации в Гейдельбергском университете. Задачи, используемые в статье, имеют точные решения и используются для выявления оптимального оператора кроссовера.

Три проблемы тестового стенда – это симметричные TSP, из них два набора данных получены из «задач реального мира»:

1. **bays29:** дорожные расстояния, связывающие 29 городов в Баварии, Германия.
2. **att48: 48** городских центров США (Падберг, Ринальди)
3. **eil101:** искусственная проблема 101 город (Эйлон, Христофидес).

Таблица 1 – Параметры тестовых данных

Имена наборов данных	bays29	att48	eil101
Количество городов	29	48	101
Оптимальная длина пути	2020	10625	629

Таблица 2 – Результаты решения задачи TSP (bays29)

	Запуск	PMX	CX	OX
Расстояние	1	2067	2241	2475
	2	2089	2198	2313
	3	2204	2271	2328
	4	2102	2263	2317
	5	2023	2120	2285
	6	2034	2174	2220
	7	2058	2083	2313
	8	2068	2187	2244
	9	2105	2143	2324
	10	2025	2138	2280

Таблица 3 – Результаты решения задачи TSP (att48)

	Запуск	PMX	CX	OX
Расстояние	1	10961	11166	11338
	2	10640	11320	11951
	3	10756	11498	11906
	4	10646	11067	11608
	5	10671	11029	11863
	6	11016	11361	12030
	7	10955	11765	12109
	8	10775	11223	11720
	9	11077	11437	11538
	10	10641	11121	11640

Таблица 4 – Результаты решения задачи TSP (eil101)

	Запуск	PMX	CX	OX
Расстояние	1	651	751	808
	2	695	711	826
	3	659	752	880
	4	642	726	802
	5	694	746	893
	6	699	694	873
	7	660	715	849
	8	637	718	858
	9	647	731	823
	10	658	741	846

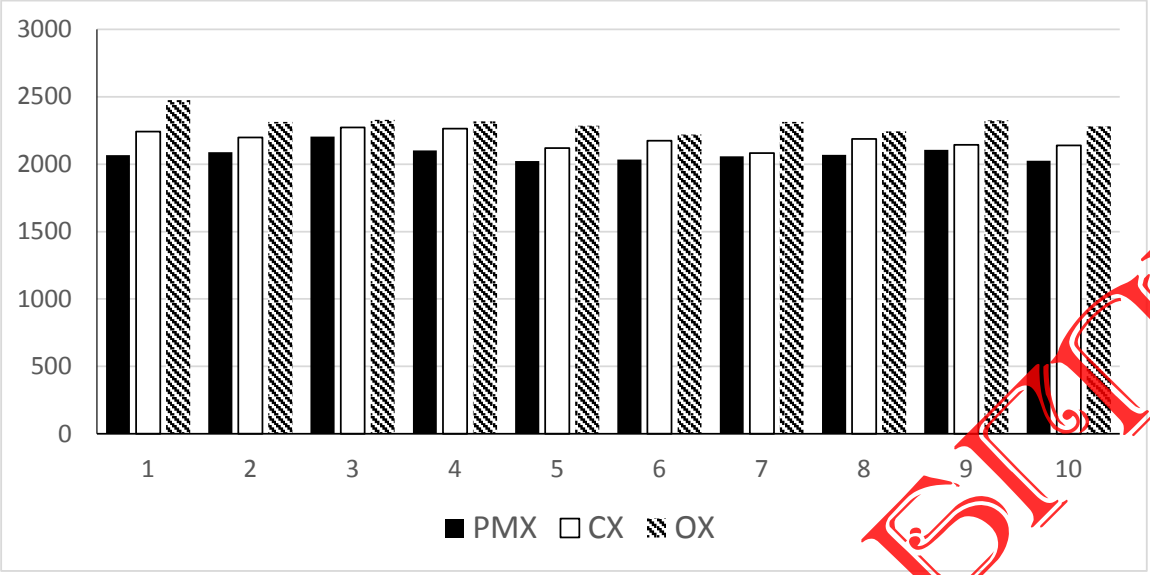

Рисунок 5 – Результаты решения задачи TSP (bays29)

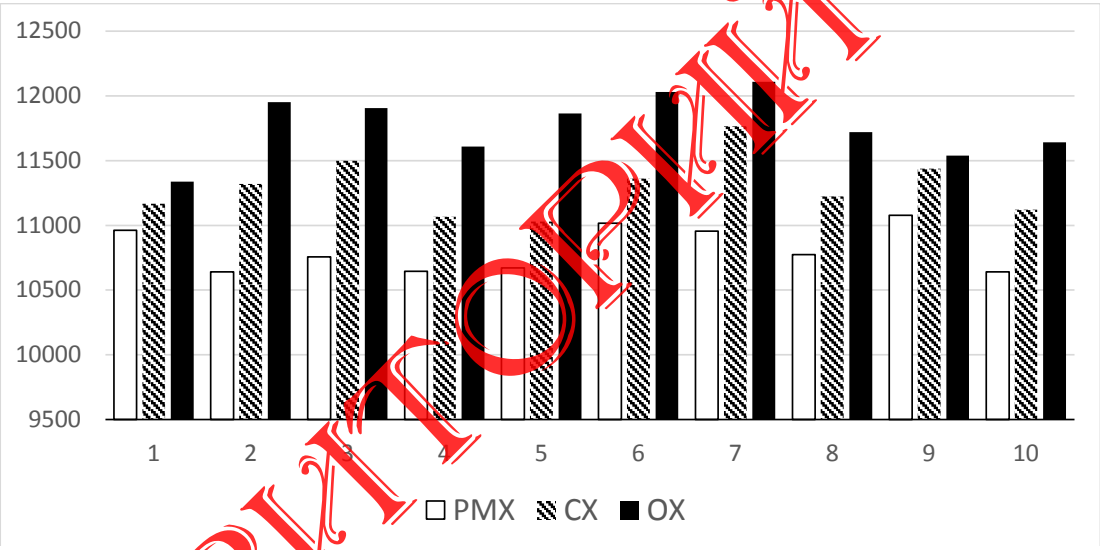

Рисунок 6 – Результаты решения задачи TSP (att48)

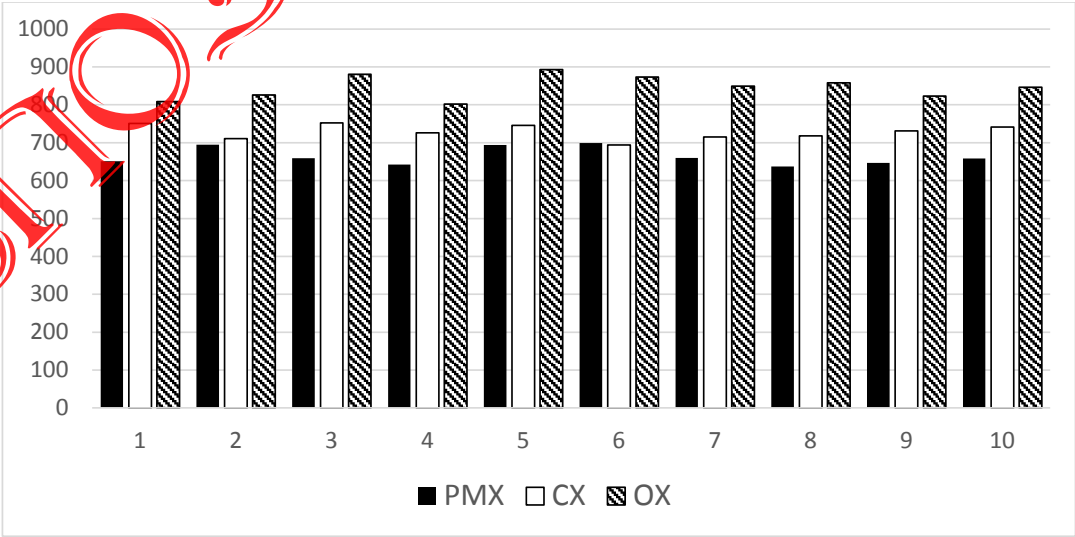

Рисунок 7 – Результаты решения задачи TSP (eil101)

Заключение. Экспериментальные результаты показывают, что оператор кроссовера PMX превосходит оператор CX и позволяет получить наилучшие значения оптимального пути по отношению к уже известному точному решению задач bays29, att48 и eil101. В свою очередь, расстояние, измеренное при использовании оператора OX, уступает по точности расстоянию, измеренному оператором CX. Таким образом, исследование показывает, что оператор кроссовера PMX превосходит операторы CX и OX.

Другие дискретные оптимизационные задачи, такие, как задачи теории расписания, решаемые генетическим алгоритмом и кодирующие решение в виде перестановок, могут извлечь выгоду при использовании наилучшего оператора кроссовера.

Литература

1. *Бураков, М. В.* Генетический алгоритм : теория и практика: учеб. пособие / М. В. Бураков. – СПб.: ГУАП, 2008. – 164 с.
2. *Панченко, Т. В.* Генетические алгоритмы : учебно-методическое пособие / Т. В. Панченко. – Астрахань : Издательский дом «Астраханский университет», 2007. – 87 с.
3. TSPLIB [Электронный ресурс]. – Режим доступа: https://wwwproxy.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/. – Дата доступа: 24.03.2018.
4. *Umbarkar, A. J.* Crossover Operators In Genetic Algorithms: A Review / A. J. Umbarkar, P. D. Sheth // Ictact journal on soft computing. – 2015. – Vol. 6. – No. 1. – pp. 1083 – 1092.

References

1. *Burakov, M. V.* Geneticheskiy algoritm : teoriya i praktika: ucheb. posobiye / M. V. Burakov. – SPb.: GUAP, 2008. – 164 s.
2. *Panchenko, T. V.* Geneticheskiye algoritmy : uchebno-metodicheskoye posobiye / T. V. Panchenko. – Astrakhan : Izdatelskiy dom “Astrakhanskiy universitet”, 2007. – 87 s.
3. TSPLIB [Elektronnyy resurs]. – Rezhim dostupa: https://wwwproxy.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/. – Data dostupa: 24.03.2018.
4. *Umbarkar, A. J.* Crossover Operators In Genetic Algorithms: A Review / A. J. Umbarkar, P. D. Sheth // Ictact journal on soft computing. – 2015. – Vol. 6. – No. 1. – Pp. 1083 – 1092.