Построение алгоритмов решения задачи коммивояжера

Жукова Екатерина Викторовна, гр. 522

Санкт-Петербургский государственный университет
Математико-механический факультет
Кафедра статистического моделирования

Научный руководитель: к.ф.-м.н. Товстик Т.М.
Рецензент: д.ф.-м.н. Ермаков С.М.

Санкт-Петербург
2012г.

Цель работы

Цель данной работы:

- в задаче коммивояжера построение алгоритмов приближенно минимизирующих длину его маршрута при посещении различного количества городов.
- нахождение алгоритма, способного уменьшить длину маршрута классического примера GR120.

Постановка задачи

Задача коммивояжера: найти кратчайший путь, чтобы обойти N городов.
Ограничения:

- путь должен быть замкнутым;
- в каждом из городов коммивояжер должен побывать точно один раз.

Цель решения: нахождение маршрута, удовлетворяющего этим условиям и при этом имеющего минимальную длину.

- $1, \ldots, N$ - узловые точки (города), $C = \{1, \ldots, N\}$ - множество узлов;
- k - количество шагов алгоритма;
- $j_i^k, 1 \leq i \leq N$ - путь обхода городов после k-ого испытания;
- $r_i^k = r(j_i^k, j_{i+1}^k)$ - расстояние между соответствующими городами.

«Маршрут» коммивояжера описывается $\varphi : C \mapsto C$, $\varphi^m(i) = j_{i+m}$ со свойствами:

- $\varphi^m(i) \neq$ i для всех $m = 1, \ldots, N-1$;
- $\varphi^N(i) = i$ для всех i, т.е. циклической перестановкой на C.

Затраты в нашем случае - это общая длина пути

$$H(k) = \sum_{i=1}^{N-1} r_i^k + r(j_N^k, j_1^k). \tag{1}$$

Предполагаем, что расстояние между городами i и j: $r(i, j) = r(j, i)$.

Постановка задачи

Выбор нового пути обхода ψ осуществляется в виде двойной замены. Представим φ в виде ориентированного графа, как на рисунке. Убираем две связи, не являющиеся соседними и выходящие из p и $\varphi^{-1}(q)$. Заменяем их на связи от p к $\varphi^{-1}(q)$ и от $\varphi(p)$ к q и меняем направление связей между $\varphi(p)$ и $\varphi^{-1}(q)$ на противоположное. Это порождает граф, изображенный на рисунке.

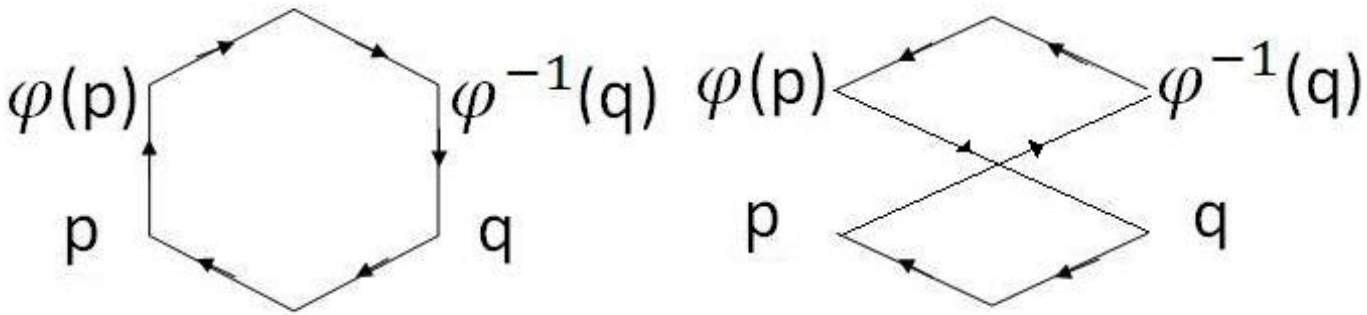

Рис. 1. Двойная замена

Будем говорить, что ψ получено из φ двойной заменой.

Алгоритмы Метрополиса и отжига

Метод Метрополиса был предложен в работе N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.N. Teller and E. Teller (1953). Через H будем обозначать функцию вычисляющую длину графа по списку обхода точек. Исходной конфигурацией будем называть порядок обхода N городов в начальный момент времени: $x^0 = (j_1^0 \to \ldots \to j_N^0)$. Новая конфигурация на k-ом шаге $x^k = (j_1^k \to \ldots \to j_N^k)$.
Текущая конфигурация x изменяется по следующему алгоритму.
1) Моделирование модификации: новая возможная конфигурация y получается с помощью двойной замены.
2) Корректировка конфигурации y (принятие или отклонения модификации):
(а) если $H(y) \leq H(x)$, то y принимается в качестве новой конфигурации;
(б) если $H(y) > H(x)$, то y принимается в качестве новой конфигурации с вероятностью $e^{-(H(y)-H(x))}$;
(в) если модификация y отклоняется, то остается старая конфигурация x.

Алгоритмы Метрополиса и отжига

Алгоритм отжига основывается на имитации физического процесса, который происходит при кристаллизации вещества, в том числе при отжиге металлов.

Алгоритм.

1) Пусть задана первоначальная конфигурация $x = x^0$. Конфигурация на k-ом шаге $x = x^k$. Переход от x^k до x^{k+1} осуществляется следующим образом.
2) Выбираем случайно две точки и соединяем их методом двойной замены. Получили конфигурацию y.
3) Если $H(y) \leq H(x)$, то y принимается в качестве новой конфигурации.
4) Если $H(y) > H(x)$, то моделируем равномерно распределенную α на $(0; 1)$.
5) Если $\alpha < e^{-\beta_k (H(y) - H(x))}$, то y принимается в качестве новой конфигурации; иначе остается старая конфигурация x.
Таким образом, в методе отжига с некоторой вероятностью, зависящей от β_k, допускается переход системы в состояние с большим расстоянием обхода.

Выбор параметра β_k

Пусть $\Delta H = H(k+1) - H(k)$. Принятие пробной конфигурации на $k+1$-ом шаге происходит с вероятностью:

$$P_{k+1} = \begin{cases} 1, & \text{если } \Delta H < 0; \\ e^{-\beta_k \Delta H}, & \text{если } \Delta H > 0. \end{cases} \tag{2}$$

Ясно, что β_k должна быть соизмерима с ΔH. При двойной замене на $k+1$ - ом шаге с $p = j_m^k, q = j_l^k$, получаем

$$\Delta H = r(p, \varphi^{-1}(q)) + r(\varphi(p), q) - r(p, \varphi(p)) - r(\varphi^{-1}(q), q). \tag{3}$$

1. **Случайное расположение исходных точек** является равномерным в единичном квадрате. Тогда

$$\mathbf{E}r = 0,5214, \quad \sigma(r) = 0,2479. \tag{4}$$

Представим β_k в виде

$$\beta_k = \frac{\hat{\beta}_k}{\sigma(\Delta H)}. \tag{5}$$

При $\Delta H > 0$ и больших k (k-число испытаний) $\hat{\beta}_k \approx \ln(k)$, тогда

$$P_{k+1} = e^{-\frac{\ln(k)\Delta H}{\sigma(\Delta H)}} = \frac{1}{k^{\frac{\Delta H}{\sigma(\Delta H)}}} \xrightarrow{k\to\infty} 0. \tag{6}$$

Методом Монте-Карло находим $\sigma(\Delta H | \Delta H > 0) = 0,2946$.

Найдем среднее значение вероятности принятия новой конфигурации, когда $\Delta H > 0$.

$$P_{k+1} = \mathbf{E}(e^{-\hat{\beta}_k \frac{\Delta H}{\sigma(\Delta H)}} | \Delta H > 0). \quad (7)$$

Связь $\hat{\beta}_k$ и P_{k+1} видно из таблицы:

Таблица: Выбор $\hat{\beta}_k$

$\hat{\beta}_k$=3	$\hat{\beta}_k$=4	$\hat{\beta}_k$=5	$\hat{\beta}_k$=6	$\hat{\beta}_k$=7
P_{k+1}=0.2712	P_{k+1}=0.2359	P_{k+1}=0.2085	P_{k+1}=0.1183	P_{k+1}=0.1121

2. **Конфигурация исходных точек задана и не является случайной.** Для маршрута после k-ого испытания найдем среднее значение $\bar{r}^k$ расстояний между городами и разброс $(\hat{\sigma}^k)^2$ по формулам

$$\bar{r}^{(k)} = \frac{1}{N}\sum r_i^{(k)}, \quad (\hat{\sigma}^{(k)})^2 = \frac{1}{N}\sum (r_i^{(k)})^2 - (\bar{r}^{(k)})^2, \quad H^{(k)} = \sum r_i^{(k)}. \quad (8)$$

Предлагается в качестве β_k использовать величину

$$\beta_k = \frac{\hat{\beta}_k}{\sqrt{(\hat{\sigma}^{(k)})^2 + (\hat{\sigma}^{(k+1)})^2}}. \quad (9)$$

С ростом числа испытаний величины $\bar{r}^{(k)}$ и $(\hat{\sigma}^{(k)})^2$ убывают (до некоторого предела), β_k в (9) должна возрастать, чтобы $P_{k+1} \xrightarrow{k\to\infty} 0$.

Построение алгоритмов

В основе алгоритма Round лежит определение полярных координат каждой точки.

1) Выбирается окружность, на которую будут перемещены точки. Эта окружность может иметь центр как в середине области, так и в любой точке области (для этого достаточно выбрать соответствующий пункт - «Random Center»).

2) Преобразование декартовых координат точек в полярные относительно выбранного центра окружности, т.к. нас интересует угол θ каждой точки:

$$\theta = \begin{cases} \arctan(\frac{y}{x}), & x > 0, y \geq 0, \\ \arctan(\frac{y}{x}) + 2\pi, & x > 0, y < 0, \\ \arctan(\frac{y}{x}) + \pi, & x < 0, \\ \frac{\pi}{2}, & x = 0, y > 0, \\ \frac{3\pi}{2}, & x = 0, y < 0, \\ 0, & x = 0, y = 0. \end{cases} \tag{10}$$

3) Упорядочивание точки по возрастанию угла θ, получая тем самым список точек на окружности, который соответствует нашему первоначальному обходу. Далее идет соединение точек согласно их расположению на окружности, по часовой стрелке, и результат этого соединения показывается на третьем шаге.

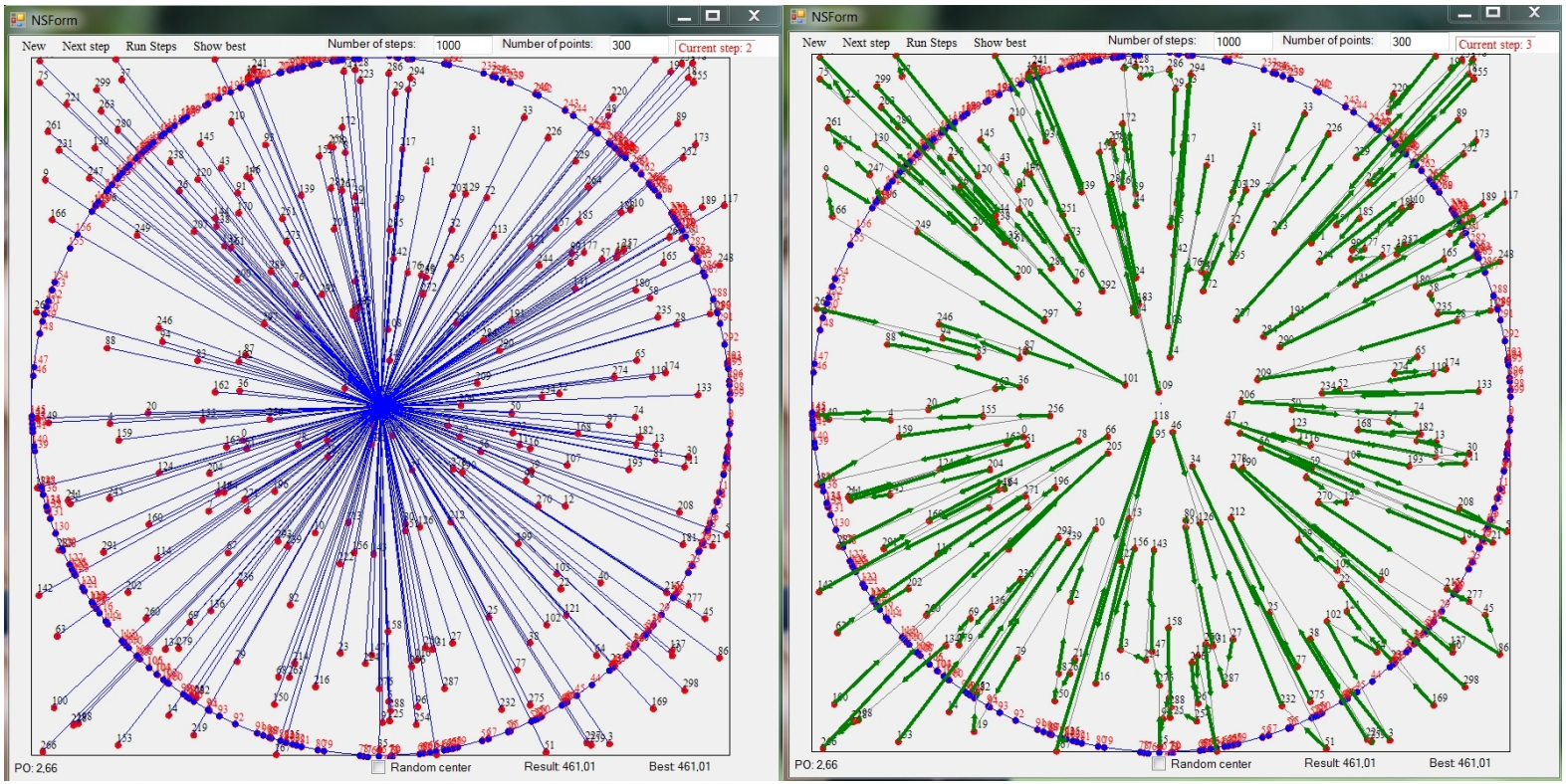

Рис. 2. Проектирование точек на окружность и «маршрут» коммивояжера

Далее рассмотрим несколько алгоритмов.

- QuadRoundSample. Область делится на четыре одинаковые подобласти. В каждой из областей проводится та же последовательность действий, что и в предыдущем алгоритме «Round», но последняя точка подобласти соединяется не с 1 точкой своей подобласти, а с 1 точкой следующей подобласти. Так мы и получаем полный обход графа.
- Quad Round Complex. Алгоритм отличается от предыдущего тем, что выполняет более сложный обход вершин графа. Обход представлен на рисунке и все так же осуществляется строго по часовой стрелке)

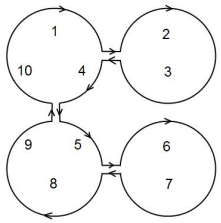

Рис. 3. Нумерация обхода в алгоритме Quad Round Complex

- Quad Round Annealing. Алгоритм аналогичен Quad Round по последовательности обхода, но в каждой из подобластей выполняется еще и метод отжига, чтобы максимально улучшить локальный результат в подобластях.
- Round Annealing. Алгоритм аналогичен Round по последовательности обхода, но в области выполняется еще и метод отжига, чтобы улучшить результат.

Алгоритм Round Annealing Complex (RAC) также делит область на четыре подобласти, но с несколько иными шагами.

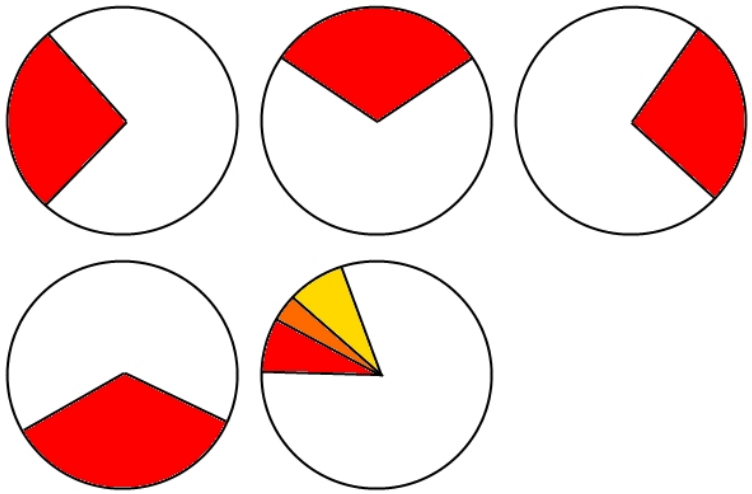

Рис. 4. Последовательность рассматриваемых областей

1) Точки упорядочиваются по расположению на одной окружности в зависимости от θ.
2) Рассматривается небольшая часть области (например, подобласть, содержащая точки с $\frac{3\pi}{4} \leq \theta \leq \frac{5\pi}{4}$ (или в количестве $\sim 70 - 80$), и в ней выполняется метод отжига.
3) Выбираются другие области, в них тоже выполняется метод отжига.
4) Проходим растром по всей окружности, находя наилучший вариант в каждой области.
Таким образом, происходят локальные улучшения результата, способствующие улучшению результата в целом.

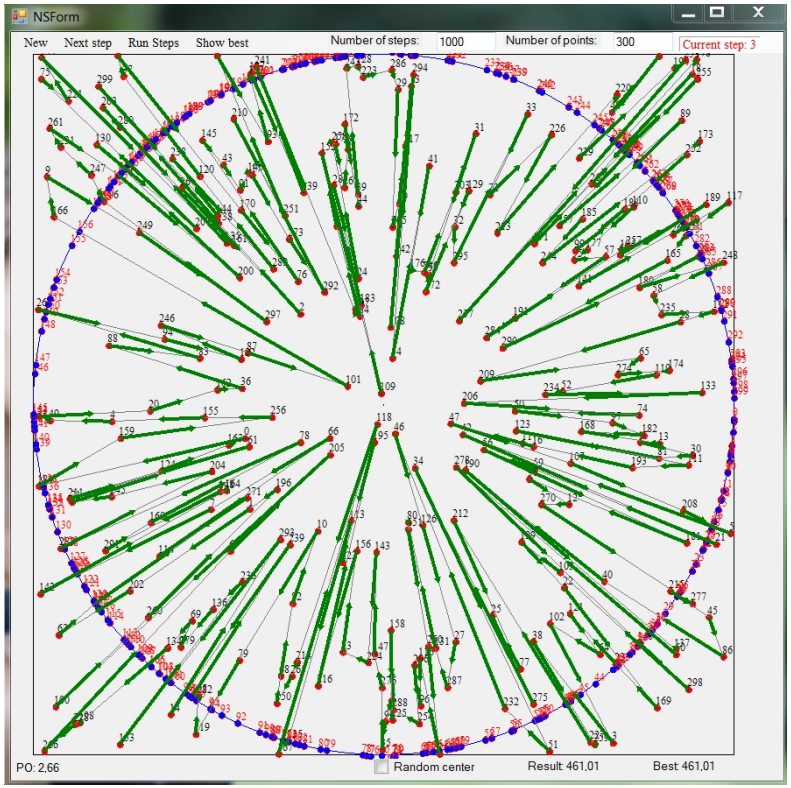

Рис. 5. Алгоритм Round, $\ell = 461,01$

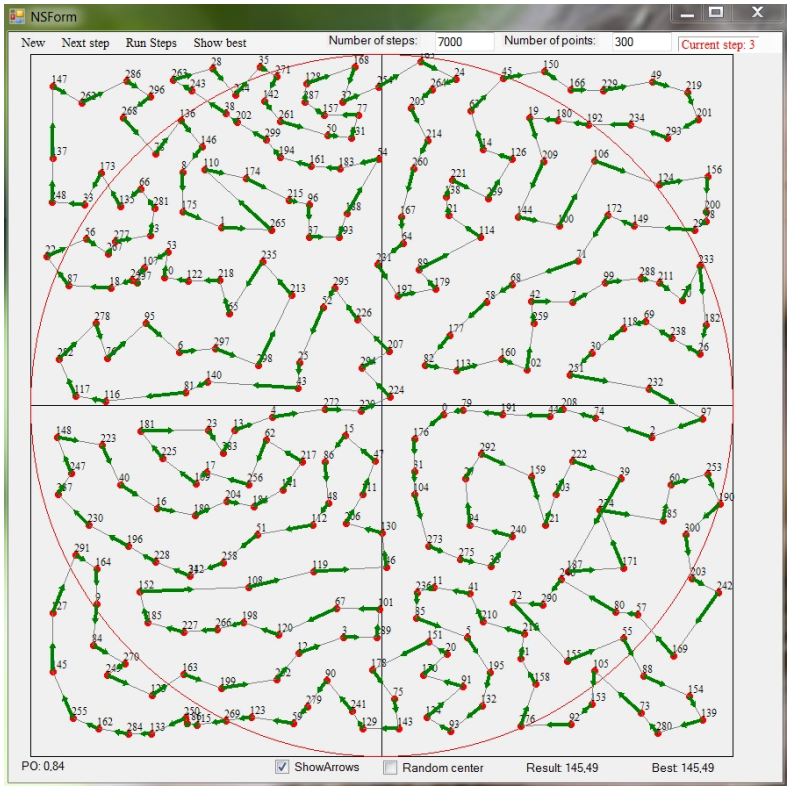

Рис. 6. Алгоритм RAC, $\ell = 145,49$

Определение

Во всех алгоритмах используется величина, называемая *нормализованной длиной маршрута*. Для маршрута φ_k эта величина определяется следующим образом:

$$\rho(\varphi_k) = \frac{H(\varphi_k)}{\sqrt{SN}}, \tag{11}$$

где S - площадь области, содержащей N городов.

Заполним таблицу полученными результатами:

Таблица: Результаты для $N = 300$

Название алгоритмов	$\ell = H(x)$ (Длина)	$\rho(\varphi_k)$
Round	461,01	2,66
Round Annealing	441,34	2,54
QuadRound Sample	229,25	1,32
Quad Round Complex	223,87	1,29
Quad Round Annealing	230,5	1,33
Round Annealing Complex	145,49	0,84

Важным результатом является то, что было проведено улучшение маршрута для классического примера GR120, наилучшим результатом (по количеству городов) полученным в 1977 году, с использованием последнего описанного алгоритма. Исходные данные были взяты из библиотеки TSPLIB.

Таблица: Результаты для $N = 120$

Алгоритм	ℓ (Длина)	ρ
M. Groetschel	1666,51	0,7459
С.М. Ермаков и С.Н. Леора	1654,76	0,7406
Новый маршрут	1650,28	0,7386

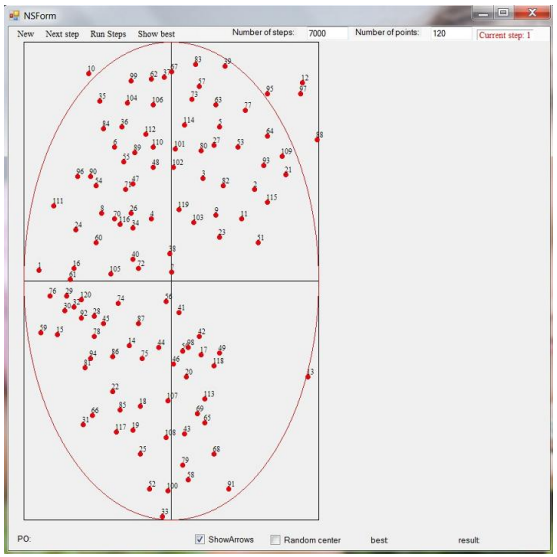

Рис. 7. Первоначальное расположение точек

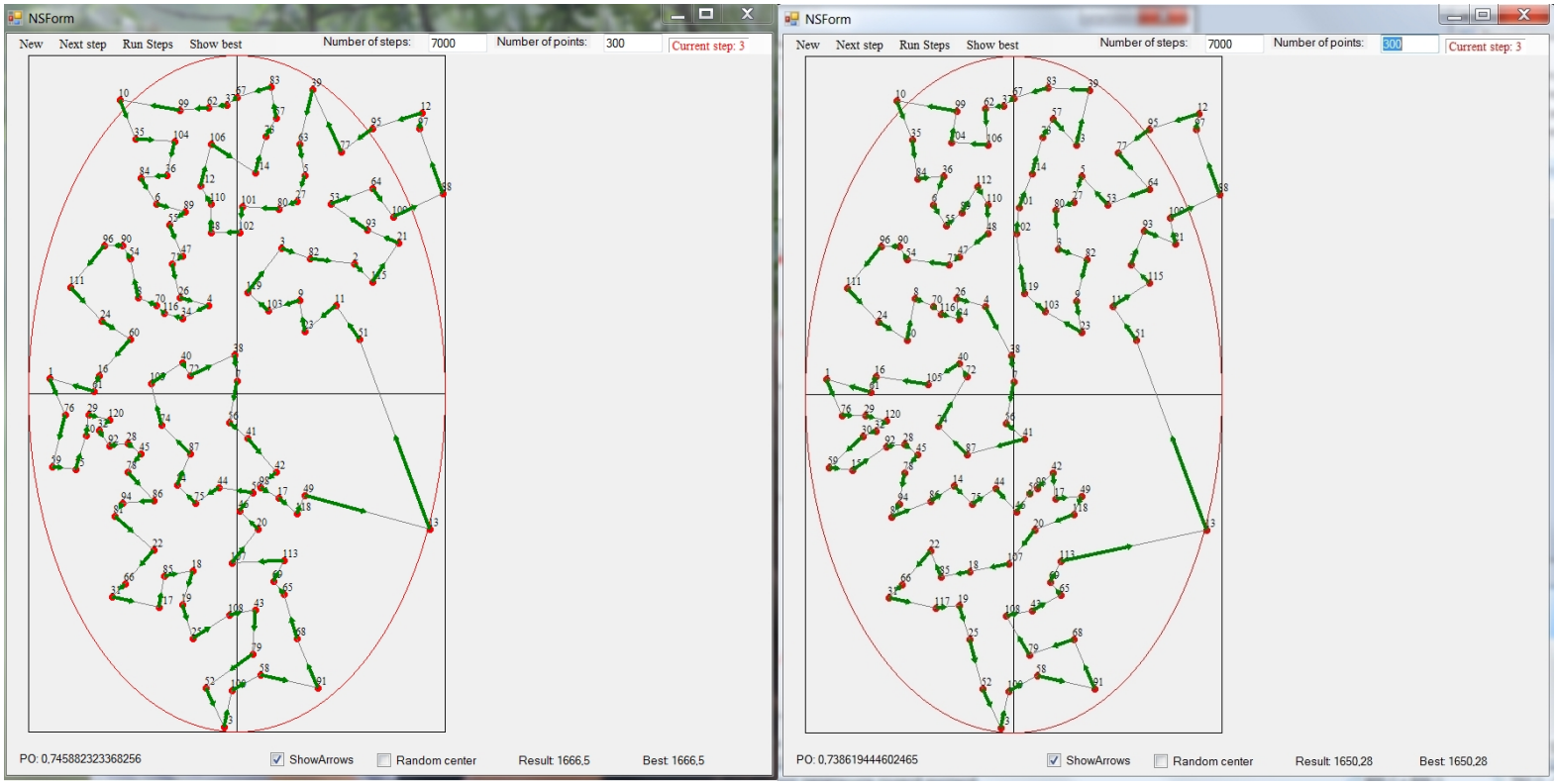

Рис. 8. «Маршрут» коммивояжера M.Groetschel и новый маршрут

Полученные результаты

- Найден алгоритм, способный уменьшить длину маршрута классического примера GR120.
- Реализованы алгоритмы и программа для приближенного решения задачи коммивояжера, удовлетворяющие всем первоначальным условиям и при этом имеющие нормализованную длину, близкую к оптимальной.
- Для реализации программы использовалась среда Microsoft Visual Studio 2010.