TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KÌ MÔN**

**PHÂN TÍCH VÀ THIẾT KẾ GIẢI THUẬT**

*Người hướng dẫn*: **GV.NGUYỄN CHÍ THIỆN**

*Người thực hiện*: **TRẦN THỊ VẸN – 52100674**

**NGUYỄN VŨ TƯỜNG – 52100944**

**VŨ MINH PHÚC – 52100921**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KÌ MÔN**

**PHÂN TÍCH VÀ THIẾT KẾ GIẢI THUẬT**

*Người hướng dẫn*: **GV.NGUYỄN CHÍ THIỆN**

*Người thực hiện*: **TRẦN THỊ VẸN – 52100674**

**NGUYỄN VŨ TƯỜNG – 52100944**

**VŨ MINH PHÚC – 52100921**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Trong suốt quá trình học tập và rèn luyện, chúng em đã nhận được rất nhiều sự giúp đỡ tận tình, sự quan tâm, chăm sóc của GV. Ngoài ra, chúng em còn được GV truyền đạt những kiến thức, phương pháp mới về toán hay ho và thú vị, thầy cô còn giúp sinh viên có được nhiều niềm vui trong việc học và cảm thấy thoải mái, … Chúng em xin chân thành cảm ơn các thầy cô rất nhiều trong suốt quá trình học tập này!

Bởi lượng kiến thức của chúng em còn hạn hẹp và gặp nhiều vấn đề trong quá trình học nên báo cáo này sẽ còn nhiều thiếu sót và cần được học hỏi thêm. Chúng em rất mong em sẽ nhận được sự góp ý của quý thầy cô về bài báo cáo này để chúng em rút kinh nghiệm trong những môn học sắp tới. Cuối cùng, chúng em xin chân thành cảm ơn quý thầy cô.

TP Hồ Chí Minh, ngày 30 tháng 11 năm 2023

Sinh viên:

TRẦN THỊ VẸN – 52100674

NGUYỄN VŨ TƯỜNG – 52100944

VŨ MINH PHÚC – 52100921

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của chúng tôi và được sự hướng dẫn của GV. Nguyễn Chí Thiện. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 30 tháng 11 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Trần Thị Vẹn*

*Nguyễn Vũ Tường*

*Vũ Minh Phúc*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Xác định vấn đề cho thuật toán (đầu vào, đầu ra theo cấu trúc dữ liệu Java). Biểu diễn Java (lập trình chung) của các thuật toán để giải quyết vấn đề mà không cần nhập thư viện bên ngoài. Phân tích độ phức tạp tiệm cận của các giải pháp(tuân thủ nghiêm ngặt các kế hoạch phân tích.)

Viết chương trình tạo tập dữ liệu đầu vào. Thử nghiệm các chương trình với các tập dữ liệu đầu vào được tạo ra. Vẽ hàm phức tạp lý thuyết và hoạt động thực tế thời gian là hàm của kích thước đầu vào. Đồ họa phải có hoa văn tương tự. Đưa ra kết luận dựa trên kết quả thực nghiệm sau khi thử nghiệm thuật toán.

MỤC LỤC

TÓM TẮT iv

MỤC LỤC v

DANH MỤC HÌNH VẼ vi

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 1

1.1 Travelling salesman problem – TSP 1

1.2 Sơ lược giải thuật trong bài báo 2

1.3 Genetic algorithm and a double-chromosome implementation 4

CHƯƠNG 2: THỰC HIỆN YÊU CẦU ĐỀ BÀI 6

2.1 Xác định vấn đề (đầu vào, đầu ra theo cấu trúc dữ liệu Java) 6

2.2 Phân tích độ phức tạp của thuật toán 7

2.3 Viết chương trình tạo tập dữ liệu đầu vào 22

2.4 Thử nghiệm các dataset được generate ra 23

2.5 Vẽ biểu đồ thời gian và số thành phố chạy 25

2.6 So sánh so với thuật toán khác: 26

TÀI LIỆU THAM KHẢO I

DANH MỤC HÌNH VẼ

Hình 1: Bản đồ đề xuất và nhiễm sắc thể hướng dẫn cho TSP 7 cities 3

Hình 2: Sơ đồ của phương pháp được đề xuất 3

Hình 3: TSP thu được bằng phương pháp đề xuất với 14 thành phố. 4

Hình 4: Tìm khoảng cách giữa các thành phố 4

Hình 5: File csv khoảng cách các thành phố 6

Hình 6: Kết quả đầu ra gồm các thông tin về thành phố số lần lặp … 6

Hình 7: Kế quả trả về final length ngắn nhất cho TSP 7

Hình 8: File data csv được generate 22

Hình 9: Kết quả thử nghiệm với 10 thành phố 23

Hình 10: Kết quả thử nghiệm với 20 thành phố 24

Hình 11: Kết quả thử nghiệm với 50 thành phố 24

Hình 12: Kết quả thử nghiệm với 100 thành phố 24

Hình 13: Thời gian chạy của 75 thành phố 25

Hình 14: Biểu đồ thời gian chạy của 75 thành phố 25

Hình 15: Thời gian chạy của 200 thành phố 26

Hình 16: Biểu đồ thời gian chạy của 200 thành phố 26

Hình 17: TSP 6 thành phố chạy bằng thuật toán Ant Colony 28

Hình 18: TSP 6 thành phố chạy bằng thuật toán Genetic 28

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

**Bài báo: Genetic algorithm and a double‑chromosome implementation to the traveling salesman problem**

Sự đa dạng của các phương pháp được sử dụng để giải quyết vấn đề người giao hàng chứng tỏ rằng vấn đề này vẫn còn phức tạp và được các nhà nghiên cứu trong lĩnh vực này quan tâm. Đối với các bài toán có không gian tìm kiếm lớn, tương tự như người bán hàng du lịch, các thuật toán tiến hóa như thuật toán di truyền rất mạnh mẽ và có thể được sử dụng để đạt được tối ưu hóa các giải pháp. Tuy nhiên, thách thức trong việc áp dụng giải thuật di truyền vào bài toán nhân viên bán hàng du lịch là việc lựa chọn các nhà khai thác thích hợp có thể sản xuất các chuyến du lịch hợp pháp. Trong tài liệu, các thuật toán sửa chữa bổ sung đã được giới thiệu và được sử dụng và con cái được tạo ra bởi những người vận hành thuật toán di truyền này được sửa đổi để đảm bảo rằng các nhiễm sắc thể được tạo ra đại diện cho các chuyến tham quan hợp pháp. Thay vì bám vào các thuật toán sửa chữa, phương pháp tiếp cận nhiễm sắc thể kép được đề xuất trong bài viết này. Phương pháp đề xuất có thể được sử dụng để tối ưu hóa bài toán tương tự như bài toán người bán hàng du lịch. Phương pháp tiếp cận nhiễm sắc thể kép đã được thử nghiệm với nhiều vấn đề khác nhau của nhân viên bán hàng du lịch và

Kết quả cho thấy phương pháp đề xuất có tốc độ hội tụ cao hướng tới hành trình ngắn nhất.

* 1. Travelling salesman problem – TSP

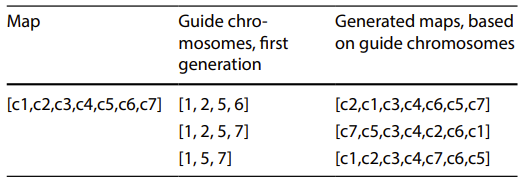
Bài toán Người du lịch Bài toán Người du lịch, tìm đường đi ngắn nhất cho người thương nhân (salesman), hay còn gọi là người chào hàng xuất phát từ một thành phố, đi qua lần lượt tất cả các thành phố duy nhất một lần và quay về thành phố ban đầu với chi phí rẻ nhất, được phát biểu vào thế kỷ 17 bởi hai nhà toán học vương quốc Anh là Sir William Rowan Hamilton và Thomas Penyngton Kirkman, và được ghi trong cuốn giáo trình Lý thuyết đồ thị nổi tiếng của Oxford. Nó nhanh chóng trở thành bài toán khó thách thức toàn thế giới bởi độ phức tạp thuật toán tăng theo hàm số mũ (trong chuyên ngành thuật toán người ta còn gọi chúng là những bài toán NP-khó). Người ta bắt đầu thử và công bố các kết quả giải bài toán này trên máy tính từ năm 1954 (49 đỉnh), cho đến năm 2004 bài toán giải được với số đỉnh lên tới 24.978, và dự báo sẽ còn tiếp tục tăng cao nữa.

Các phương pháp giải quyết bài toán TSP bao gồm:

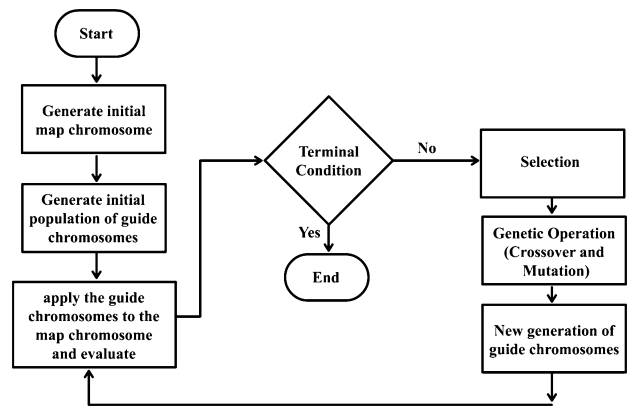
* Brute Force: Tính toán tất cả các hoán vị của các điểm và chọn hoán vị tối ưu. Tuy nhiên, phương pháp này không thực tế cho các bài toán lớn vì số lượng hoán vị là rất lớn.
* Nearest Neighbor: Bắt đầu từ một điểm bất kỳ, tìm điểm gần nhất và di chuyển đến nó, sau đó tìm điểm gần nhất với điểm đang đứng và tiếp tục cho đến khi tất cả các điểm được ghé thăm. Phương pháp này dễ triển khai và thường được sử dụng cho các bài toán nhỏ.
* Christofides Algorithm: Kết hợp giữa Nearest Neighbor và Minimum Spanning Tree để tìm một lời giải gần đúng với tỷ lệ gần 3/2 so với giải tối ưu. Phương pháp này được sử dụng cho các bài toán lớn.
* Simulated Annealing: Một phương pháp metaheuristics, nó tạo ra một lời giải bắt đầu ngẫu nhiên và sau đó cố gắng tìm một lời giải tốt hơn bằng cách di chuyển qua các lời giải khác theo một quy trình tương tự như quá trình làm mát trong hành trình.
* Genetic Algorithm: Một thuật toán tối ưu hóa được lấy cảm hứng từ các quá trình tiến hóa trong tự nhiên. Thuật toán này sử dụng một quần thể các lời giải và tiến hành các phép lai ghép, đột biến để tìm ra lời giải tốt hơn và nhiều thuật toán khác.
  1. Sơ lược giải thuật trong bài báo

Trong một TSP đối xứng trong đó khoảng cách giữa hai thành phố là như nhau theo mỗi hướng ngược nhau, tổng số chuyến tham quan có thể có (θ) của n thành phố có thể thu được bằng cách loại bỏ các chuyến tham quan tương tự như sau:

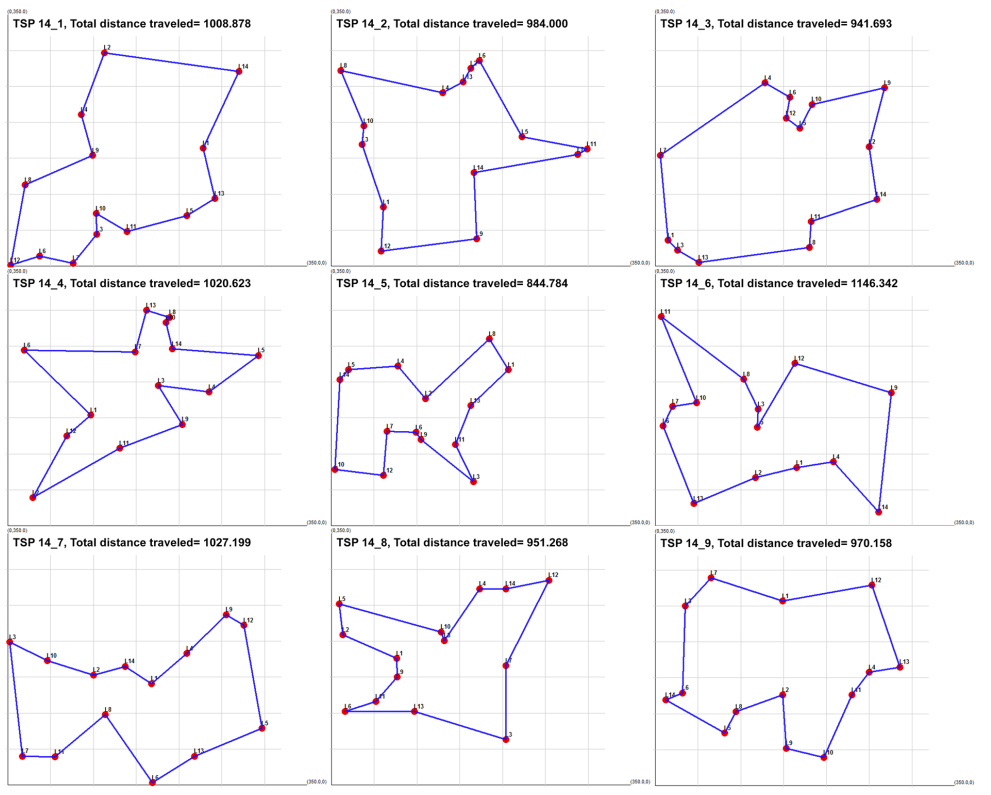




Hình 1: Bản đồ đề xuất và nhiễm sắc thể hướng dẫn cho TSP 7 cities



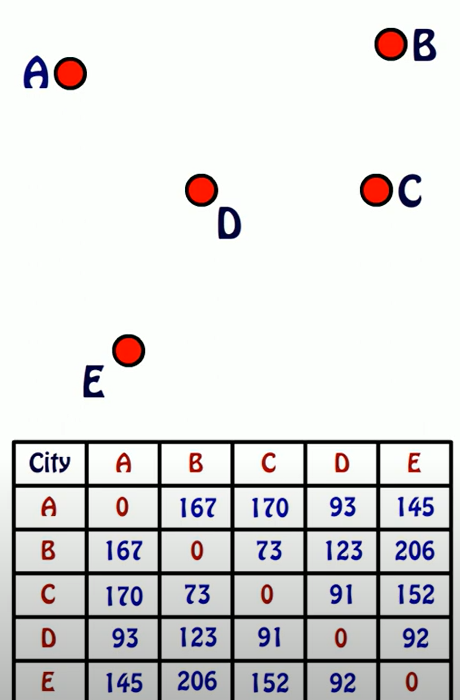
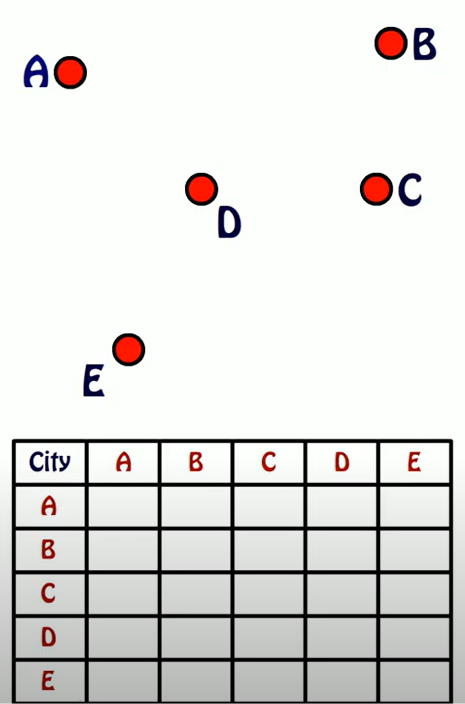
Hình 2: Sơ đồ của phương pháp được đề xuất



Hình 3: TSP thu được bằng phương pháp đề xuất với 14 thành phố.

* 1. Genetic algorithm and a double-chromosome implementation

Bước 1: Tìm khoảng cách giữa các thành phố



Hình 4: Tìm khoảng cách giữa các thành phố

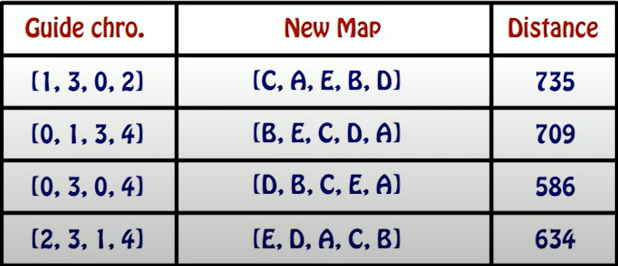
Bước 2: Tạo ra một random Map chromosome



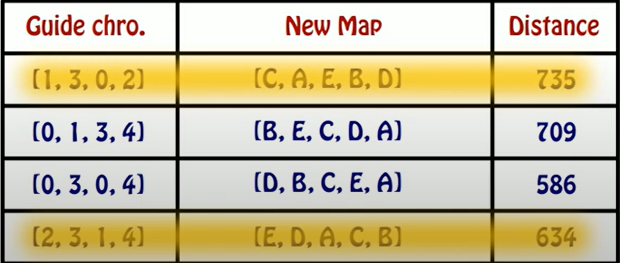
Bước 3: Tạo ra random Guide chromosome

Các nhiễm sắc thể dẫn đường được sử dụng để sắp xếp lại nhiễm sắc thể bản đồ. Trong các nhiễm sắc thể hướng dẫn được đề xuất, mỗi cặp gen sẽ xác định hai thành phố nào trong nhiễm sắc thể bản đồ sẽ được hoán đổi.

Do đó, các nhiễm sắc thể dẫn đường hoạt động theo cách sắp xếp nhiễm sắc thể bản đồ để đạt được hành trình ngắn nhất.



Bước 4: Selection (roulette-wheel)





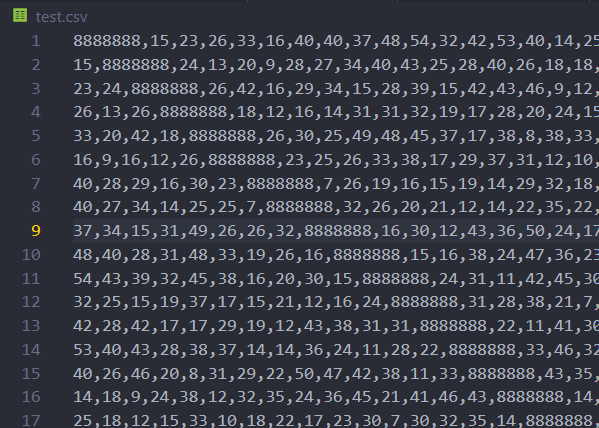
Bước 5: Uniform Crossover



CHƯƠNG 2: THỰC HIỆN YÊU CẦU ĐỀ BÀI

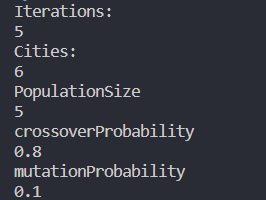
* 1. Xác định vấn đề (đầu vào, đầu ra theo cấu trúc dữ liệu Java)

Đầu vào:

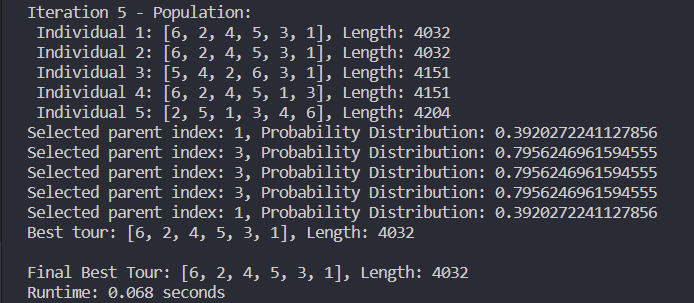


Hình 5: File csv khoảng cách các thành phố

Đầu ra:



Hình 6: Kết quả đầu ra gồm các thông tin về thành phố số lần lặp …



Hình 7: Kế quả trả về final length ngắn nhất cho TSP

* 1. Phân tích độ phức tạp của thuật toán

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A – 1 | B – 2 | C – 3 | D – 4 | E – 5 | F – 6 |
| A – 1 | 0 | 1930 | 1484 | 1096 | 991 | 1152 |
| B – 2 | 1930 | 0 | 684 | 1091 | 1211 | 1582 |
| C – 3 | 1484 | 684 | 0 | 459 | 1427 | 884 |
| D – 4 | 1096 | 1091 | 459 | 0 | 1046 | 808 |
| E – 5 | 991 | 1211 | 1427 | 1046 | 0 | 1619 |
| F – 6 | 1152 | 1582 | 884 | 808 | 1619 | 0 |

Perform with parameters:

Iterations of genetic algorithm: 4

Population size of : 5

Single point crossover with crossover probability: 0.8

Swap mutation with probability: 0.1

Different paths: 1 -> 2 -> 3 -> 4 ->5 -> 6 ->1 (Random solutions)

**Step 1: Initial Polutation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 5 Random Population | Path length | Fitness score | Fitness rank |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 | 459+808+1619+1211 | 4097 | 2 |
| Parent 2 | 2 -> 4-> 3-> 5-> 6 | 1091+459+1427+1619 | 4596 | 4 |
| Parent 3 | 4 -> 6-> 5-> 3-> 2 | 808+1619+1427+684 | 4538 | 3 |
| Parent 4 | 2 -> 6-> 5-> 3-> 4 | 1582+1619+1427+459 | 5087 | 5 |
| Parent 5 | 6 -> 4-> 3-> 2-> 5 | 808+459+684+1211 | 3162 | 1 |
|  | Chromosomes | Fitness function | Fitness value |  |
|  |  | Total fitness score: | 21480 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Random Population | Fitness score | Relative Fitness | Probability of selection |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 | 4097 | 21480/4097 = 5.24286 | 5.24286/25.66552 = 0.2043 |
| Parent 2 | 2 -> 4-> 3-> 5-> 6 | 4596 | 21480/4596 = 4.67362 | 4.67362/25.66552 = 0.1821 |
| Parent 3 | 4 -> 6-> 5-> 3-> 2 | 4538 | 21480/4538 = 4.73336 | 4.73336/25.66552 = 0.1844 |
| Parent 4 | 2 -> 6-> 5-> 3-> 4 | 5087 | 21480/5087 = 4.22252 | 4.22252/25.66552 = 0.1645 |
| Parent 5 | 6 -> 4-> 3-> 2-> 5 | 3162 | 21480/3162 = 6.79316 | 6.79316/25.66552 = 0.2647 |
| Sum of relative fitness | | | 25.66552 |  |

**Step 2: Parent Selection:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Random Population | Fitness score | Probability of selection | Probability distribution |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 | 4097 | 5.24286/25.66552 = 0.2043 | 0.2043 |
| Parent 2 | 2 -> 4-> 3-> 5-> 6 | 4596 | 4.67362/25.66552 = 0.1821 | 0.3864 |
| Parent 3 | 4 -> 6-> 5-> 3-> 2 | 4538 | 4.73336/25.66552 = 0.1844 | 0.5708 |
| Parent 4 | 2 -> 6-> 5-> 3-> 4 | 5087 | 4.22252/25.66552 = 0.1645 | 0.7353 |
| Parent 5 | 6 -> 4-> 3-> 2-> 5 | 3162 | 6.79316/25.66552 = 0.2647 | 1 |

Select Random number = 0.826

Parent 5 has cumulative probability > random number

* Parent 5 is selected as first parent

Select Random number = 0.094

Parent 1 has cumulative probability > random number

* Parent 1 is selected as second parent

**Step 3: Reproduction:**

|  |  |
| --- | --- |
| Parent 5 | 6 -> 4 -> 3 -> 2 -> 5 |
| Parent 1 | 3 -> 4 -> 6 -> 5 -> 2 |

|  |  |
| --- | --- |
| Parent 1 | 6 -> 4 -> 3 -> 2 -> 5 |
| Parent 2 | 3 -> 4 -> 6 -> 5 -> 2 |

Apply single point crossover with probability 0.8, we generate a random number to decide if crossover it to be done or not.

Random number <= crossover probability then select crossover point randomly at any place in parent string.

Random number : 0.561 <=0.8

Perform crossover!

|  |  |
| --- | --- |
| Parent 1 | 6 -> 4 | 3 -> 2-> 5 |
| Parent 2 | 3 -> 4 | 6 -> 5-> 2 |

|  |  |
| --- | --- |
| Parent 1 | 6 -> 4 | 3-> 2-> 5 |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |

To avoid violation of TSP we perform swap in Parent 1

|  |  |
| --- | --- |
| Parent 1 | 3 -> 4 | 6-> 2-> 5 |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |

Swapping done to avoid populating a member twice and volating the TSP

|  |  |
| --- | --- |
| Parent 1 | 3 -> 4 | 6-> 2-> 5 |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |

Successfully transferred 1st member from P2 to P1

You can be transferred to P1 from P2 without swapping needed.

So first offspring or child: 3 -> 4 -> 6 -> 2-> 5

|  |  |
| --- | --- |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |
| Parent 1 | 6 -> 4 | 3-> 2-> 5 |

|  |  |
| --- | --- |
| Parent 2 | 6 -> 4 | 3-> 5-> 2 |
| Parent 1 | 6 -> 4 | 3-> 2-> 5 |

So second offspring or child: 6 -> 4 -> 3 -> 5-> 2

|  |  |
| --- | --- |
|  | Random Population |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 |
| Parent 2 | 2 -> 4-> 3-> 5-> 6 |
| Parent 3 | 4 -> 6-> 5-> 3-> 2 |
| Parent 4 | 2 -> 6-> 5-> 3-> 4 |
| Parent 5 | 6 -> 4-> 3-> 2-> 5 |
| Child 1 | 3 -> 4-> 6-> 2-> 5 |
| Child 2 | 6 -> 4-> 3-> 5-> 2 |

**Step 4: Mutation:**

Random Number = 0.273 > 0.1 (Probability of mutation)

But mutation not be done on the two offsprings found.

Step 5: Replacement:

1. Evalution of total tour distance of initial population and the two offsprings
2. Select best perform individuals using (preservation with selection pressure 10%)

|  |  |  |
| --- | --- | --- |
|  | Random Polutation | Total length |
| Parent 1 | 1 -> 3 -> 4-> 6-> 5 -> 2 -> 1 | 4097 + 1484 + 1930 = 7511 |
| Parent 2 | 1 -> 2 -> 4-> 3-> 5 -> 6 -> 1 | 4596 + 1930 + 1152 = 7678 |
| Parent 3 | 1 -> 4 -> 6-> 5-> 3 -> 2 -> 1 | 4538 + 1096 + 1930 = 7564 |
| Parent 4 | 1 -> 2 -> 6-> 5-> 3 -> 4 -> 1 | 5087 + 1930 + 1096 = 8113 |
| Parent 5 | 1 -> 6 -> 4-> 3-> 2 -> 5 -> 1 | 3162 + 1152 + 991 = 5305 |
| C1 | 1 -> 3 -> 4 -> 6 -> 2-> 5 ->1 | 1484 + 459 + 808 + 1582 + 1211 + 991 = 6535 |
| C2 | 1 -> 6 -> 4 -> 3 -> 5-> 2 ->1 | 1152 + 808 + 459 + 1427 + 1211 + 1930 = 6987 |

Random number: 0.011

Choose Parent 1 because 0.2043 > 0.011 (random number)

Random number: 0.723

Choose Parent 4 because 0.753 > 0.723 (random number)

Parent selected:

|  |  |
| --- | --- |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 |
| Parent 4 | 2 -> 6-> 5-> 3-> 4 |

|  |  |
| --- | --- |
| Parent 1 | 3 -> 4-> 6-> 5-> 2 |
| Parent 2 | 2 -> 6-> 5-> 3-> 4 |

Random number: 0.594 < 0.8 Apply crossover

|  |  |
| --- | --- |
| Parent 1 | 3 -> 4 | 6-> 5-> 2 |
| Parent 2 | 2 -> 6 | 5-> 3-> 4 |

|  |  |
| --- | --- |
| Parent 1 | 2 -> 4 | 6-> 5-> 3 |
| Parent 2 | 2 -> 6 | 5-> 3-> 4 |

|  |  |
| --- | --- |
| Parent 1 | 2 -> 6 | 4-> 5-> 3 |
| Parent 2 | 2 -> 6 | 5-> 3-> 4 |

New child 3: 2 -> 6 -> 4 -> 5 -> 3

|  |  |
| --- | --- |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |
| Parent 1 | 2 -> 6 | 5-> 3-> 4 |

|  |  |
| --- | --- |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |
| Parent 1 | 3 -> 6 | 5-> 2-> 4 |

|  |  |
| --- | --- |
| Parent 2 | 3 -> 4 | 6-> 5-> 2 |
| Parent 1 | 3 -> 4 | 5-> 2-> 6 |

Child 4: 3 -> 4 -> 5-> 2-> 6

Mutation:

Random number: 0.607 > 0.1 Mutation Probability

Skip mutation

|  |  |  |
| --- | --- | --- |
|  | Random Polutation | Total length |
| Parent 1 | 1 -> 3 -> 4-> 6-> 5 -> 2 -> 1 | 4097 + 1484 + 1930 = 7511 |
| Parent 2 | 1 -> 2 -> 4-> 3-> 5 -> 6 -> 1 | 4596 + 1930 + 1152 = 7678 |
| Parent 3 | 1 -> 4 -> 6-> 5-> 3 -> 2 -> 1 | 4538 + 1096 + 1930 = 7564 |
| Parent 4 | 1 -> 2 -> 6-> 5-> 3 -> 4 -> 1 | 5087 + 1930 + 1096 = 8113 |
| Parent 5 | 1 -> 6 -> 4-> 3-> 2 -> 5 -> 1 | 3162 + 1152 + 991 = 5305 |
| C1 | 1 -> 3 -> 4 -> 6 -> 2-> 5 ->1 | 1484 + 459 + 808 + 1582 + 1211 + 991 = 6535 |
| C2 | 1 -> 6 -> 4 -> 3 -> 5-> 2 ->1 | 1152 + 808 + 459 + 1427 + 1211 + 1930 = 6987 |
| C3 | 1 -> 2 -> 6 -> 4 -> 5 -> 3->1 | 1930 + 1582 + 808 + 1046 + 459 + 1484 = 7309 |
| C4 | 1 -> 3 -> 4 -> 5-> 2 -> 6 ->1 | 1484 + 459 + 1046 + 1211 + 1582 + 1152 = 6934 |

Interation 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Random Polutation | Fitness score | Relative Fitness | Probability of selection | Probability distribution |
| Parent 5 | 6 -> 4 -> 3-> 2 -> 5 | 3162 | 20288/3162 = 6.416 | 0.252 | 0.252 |
| C1 | 3 -> 4 -> 6 -> 2-> 5 | 4060 | 20288/4060 = 4.997 | 0.196 | 0.448 |
| C2 | 6 -> 4 -> 3 -> 5-> 2 | 3905 | 20288/3905 = 5.195 | 0.204 | 0.652 |
| C3 | 2 -> 6 -> 4 -> 5 ->3 | 4863 | 20288/4863 = 4.172 | 0.164 | 0.916 |
| C4 | 3 -> 4 -> 5-> 2 -> 6 | 4298 | 20288/4298 = 4.720 | 0.185 | 1.00 |
| Total Fitness: | | | 20288 |  |  |

Random number: 0.409 choose C1: 0.448

Random number: 0.525 choose C2: 0.652

Select 2 parent:

|  |  |
| --- | --- |
| C1 | 3 -> 4 -> 6 -> 2-> 5 |
| C2 | 6 -> 4 -> 3 -> 5-> 2 |

Random number: 0.336 <= 0.8 crossover

|  |  |
| --- | --- |
| P1 | 3 -> 4 | 6 -> 2-> 5 |
| P2 | 6 -> 4 | 3 -> 5-> 2 |

|  |  |
| --- | --- |
| P1 | 6 -> 4 | 3 -> 2-> 5 |
| P2 | 6 -> 4 | 3 -> 5-> 2 |

Child 5: 6 -> 4 -> 3 -> 2 -> 5

|  |  |
| --- | --- |
| P1 | 3 -> 4 | 6 -> 2-> 5 |
| P2 | 6 -> 4 | 3 -> 5-> 2 |

|  |  |
| --- | --- |
| P1 | 3 -> 4 | 6 -> 2-> 5 |
| P2 | 3 -> 4 | 6 -> 5-> 2 |

Child 6: 3 -> 4 -> 6 -> 5 -> 2

Random number: 0.53 > 0.1

Skip mutation

Tour length:

|  |  |  |
| --- | --- | --- |
|  | Random Polutation | Total length |
| Parent 1 | 1 -> 3 -> 4-> 6-> 5 -> 2 -> 1 | 4097 + 1484 + 1930 = 7511 |
| Parent 2 | 1 -> 2 -> 4-> 3-> 5 -> 6 -> 1 | 4596 + 1930 + 1152 = 7678 |
| Parent 3 | 1 -> 4 -> 6-> 5-> 3 -> 2 -> 1 | 4538 + 1096 + 1930 = 7564 |
| Parent 4 | 1 -> 2 -> 6-> 5-> 3 -> 4 -> 1 | 5087 + 1930 + 1096 = 8113 |
| Parent 5 | 1 -> 6 -> 4-> 3-> 2 -> 5 -> 1 | 3162 + 1152 + 991 = 5305 |
| C1 | 1 -> 3 -> 4 -> 6 -> 2-> 5 ->1 | 1484 + 459 + 808 + 1582 + 1211 + 991 = 6535 |
| C2 | 1 -> 6 -> 4 -> 3 -> 5-> 2 ->1 | 1152 + 808 + 459 + 1427 + 1211 + 1930 = 6987 |
| C3 | 1 -> 2 -> 6 -> 4 -> 5 -> 3->1 | 1930 + 1582 + 808 + 1046 + 459 + 1484 = 7309 |
| C4 | 1 -> 3 -> 4 -> 5-> 2 -> 6 ->1 | 1484 + 459 + 1046 + 1211 + 1582 + 1152 = 6934 |
| C5 | 1 -> 6 -> 4 -> 3 -> 2 -> 5->1 | 5305 |
| C6 | 1-> 3 -> 4 -> 6 -> 5 -> 2-> 1 | 7511 |

C5 = P5 (Polution2)

C6 = P1 (Polution1)

New offspring polutation:

Random number: 0.392 choose C1 as parent

Random number: 0.352 choose C2 as parent

Random number: 0.692 choose C3 as parent

Random number: 0.734 choose C4 as parent

New parent C3,C4

Random number: 0.245 < 0.8 crossover

|  |  |
| --- | --- |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |
| C4 | 3 -> 4 -> 5 -> 2 -> 6 |

|  |  |
| --- | --- |
| P1 | 2 -> 6 | 4 -> 5 -> 3 |
| P2 | 3 -> 4 | 5 -> 2 -> 6 |

|  |  |
| --- | --- |
| P1 | 3 -> 4 | 6 -> 5 -> 2 |
| P2 | 3 -> 4 | 5 -> 2 -> 6 |

New child but like P1 : 3 -> 4 -> 6 -> 5 -> 2

|  |  |
| --- | --- |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |
| C4 | 3 -> 4 -> 5 -> 2 -> 6 |

|  |  |
| --- | --- |
| P2 | 2 -> 6 | 4 -> 5 -> 3 |
| P1 | 3 -> 4 | 5 -> 2 -> 6 |

|  |  |
| --- | --- |
| P2 | 2 -> 6 | 4 -> 5 -> 3 |
| P1 | 2 -> 6 | 5 -> 3 -> 4 |

New child but like P4: 2 -> 6 -> 5 -> 3 -> 4

No new offspring found

Random number: 0.367 choose C1

Random number: 0.760 choose C3

New parent C1 và C3

Random number: 0.564 < 0.8 crossover

|  |  |
| --- | --- |
| C1 | 3 -> 4 -> 6 -> 2-> 5 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

|  |  |
| --- | --- |
| C1 | 2 -> 6 -> 4 -> 3-> 5 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

Child 7: 2 -> 6 -> 4 -> 3-> 5

Total length : 7197

|  |  |
| --- | --- |
| C1 | 3 -> 4 -> 6 -> 2-> 5 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

|  |  |
| --- | --- |
| C1 | 3 -> 4 -> 6 -> 2-> 5 |
| C3 | 3 -> 4 -> 6 -> 5 -> 2 |

Child 8 but like P1

Random number: 0.392 choose C1 as parent

Random number: 0.644 choose C2 as parent

Random number: 0.724 choose C3 as parent

Random number: 0.967 choose C4 as parent

**Random number: 0.630 choose C2 as parent**

**Random number: 0.102 choose P5 as parent**

Random number: 0.366 < 0.8 => crossover to follow

|  |  |
| --- | --- |
| C2 | 6 -> 4 | 3-> 5-> 2 |
| Parent 5 | 6 -> 4 | 3-> 2 -> 5 |

6 -> 4 -> 3-> 5-> 2 => Nothing new

|  |  |
| --- | --- |
| Parent 5 | 6 -> 4 | 3-> 2 -> 5 |
| C2 | 6 -> 4 | 3-> 5-> 2 |

6 -> 4-> 3-> 2 -> 5 => Nothing new

Crossover again:

|  |  |
| --- | --- |
| C2 | 6 -> 4 -> 3-> 5 | 2 |
| Parent 5 | 6 -> 4-> 3-> 2 | 5 |

6 -> 4 -> 3-> 2 -> 5 => Nothing new

|  |  |
| --- | --- |
| Parent 5 | 6 -> 4-> 3-> 2 | 5 |
| C2 | 6 -> 4 -> 3-> 5 | 2 |

6 -> 4 -> 3-> 5 ->2=> Nothing new

Random number: 0.072 choose P5 as parent

Random number: 0.508 choose C2 as parent

Random number: 0.499 choose C2 as parent

Random number: 0.035 choose P5 as parent

**Random number: 0.072 choose P5 as parent**

**Random number: 0.922 choose C4 as parent**

**Random number = 0.694 < 0.8 crossover**

|  |  |
| --- | --- |
| Parent 5 | 6 -> 4-> 3-> 2 -> 5 |
| C4 | 3 -> 4 -> 5-> 2 -> 6 |

**New:** 3 -> 4-> 6-> 2 -> 5 same C1

|  |  |
| --- | --- |
| C4 | 3 -> 4 -> 5-> 2 -> 6 |
| Parent 5 | 6 -> 4-> 3-> 2 -> 5 |

**New child C8:** 6 -> 4 -> 5-> 2 -> 3

Total length: 6385

Random number: 0.274 > 0.1 no mutation

Till now:

|  |  |  |
| --- | --- | --- |
|  | Total length |  |
| Parent 5 | 5305 |  |
| C1 | 6535 |  |
| C2 | 6987 |  |
| C3 | 7309 |  |
| C4 | 6934 |  |
| C5 | 5305 |  |
| C6 | 7511 |  |
| C7 | 7197 | Replace C3 |
| C8 | 6385 | Replace C2 |

Random number: 0.929 choose C4 as parent

Random number: 0.790 choose C3 as parent

Random number: 0.940 choose C4 as parent

Random number: 0.710 choose C3 as parent

Random number: 0.314 choose C1 as parent

Random number: 0.361 choose C2 as parent

**Random number: 0.362 choose C2 as parent**

**Random number: 0.617 choose C3 as parent**

Random number: 0.769 < 0.8 => crossover

|  |  |
| --- | --- |
| C2 | 6 -> 4 -> 3 -> 5-> 2 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

|  |  |
| --- | --- |
| C2 | 6 -> 4 | 3 -> 5-> 2 |
| C3 | 2 -> 6 | 4 -> 5 -> 3 |

New child: 2 -> 4 -> 3 -> 5-> 6

Total: 7965 (unfit)

|  |  |
| --- | --- |
| C3 | 2 -> 6 | 4 -> 5 -> 3 |
| C2 | 6 -> 4 | 3 -> 5-> 2 |

New child C9: 6 -> 4 -> 2 -> 5-> 3

Total: 7173 Replace C7

Random number: 0.756 > 0.1 no mutation

Random number: 0.417 choose C1 as parent

Random number: 0.638 choose C2 as parent

**Random number: 0.298 choose P5 as parent**

**Random number: 0.687 choose C3 as parent**

Random number: 0.652 <0.8 crossover

|  |  |
| --- | --- |
| Parent 5 | 6 -> 4-> 3 -> 2 -> 5 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

|  |  |
| --- | --- |
| Parent 5 | 2 ->4 -> 3 -> 6 -> 5 |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |

New child C10: 2 ->6 -> 3 -> 4 -> 5

Total length: 6892

|  |  |
| --- | --- |
| C3 | 2 -> 6 -> 4 -> 5 -> 3 |
| Parent 5 | 6 -> 4-> 3 -> 2 -> 5 |

|  |  |
| --- | --- |
| C3 | 6 -> 2 -> 4 -> 5 -> 3 |
| Parent 5 | 6 -> 4 -> 3 -> 2 -> 5 |

Nothing child: 6 -> 4 -> 2 -> 5 -> 3 same C9

Total length: 7173

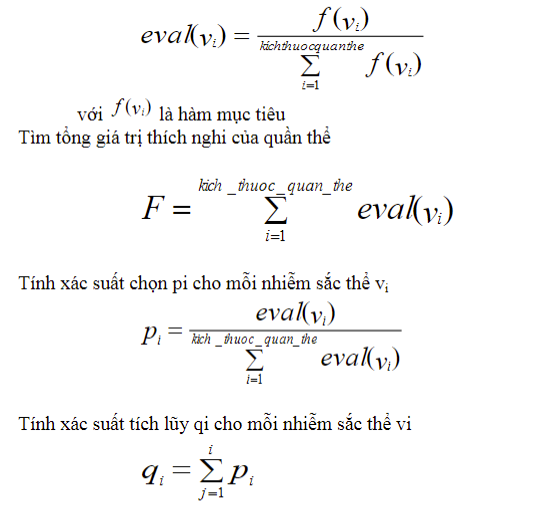
|  |  |  |
| --- | --- | --- |
|  | Total length |  |
| Parent 5 | 5305 |  |
| C1 | 6535 |  |
| C2 | 6987 |  |
| C3 | 7309 |  |
| C4 | 6934 |  |
| C5 | 5305 |  |
| C6 | 7511 |  |
| C7 | 7197 | Replace C3 |
| C8 | 6385 | Replace C2 |
| C9 | 7173 |  |
| C10 | 6892 |  |

Iteration 3:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Random Polutation | Fitness score | Relative Fitness | Probability of selection | Probability distribution | Length |
| P5 | 6 -> 4 -> 3-> 2 -> 5 | 3162 | 19717/3162 = 6.236 | 0.246 | 0.246 | 5305 |
| C8 | 6 -> 4 -> 5-> 2 -> 3 | 3749 | 19717/3749 = 5.259 | 0.207 | 0.453 | 6385 |
| C10 | 2 -> 6 -> 3-> 4 -> 5 | 3971 | 19717/3971 = 4.965 | 0.196 | 0.649 | 6892 |
| C9 | 6 -> 4 -> 2 -> 5-> 3 | 4537 | 19717/4863 = 4.346 | 0.171 | 0.820 | 7173 |
| C4 | 3 -> 4 -> 5-> 2 -> 6 | 4298 | 19717/4298 = 4.587 | 0.181 | 1.00 | 6934 |
| Total Fitness: | | | 19717 |  |  |  |

Iteration 4:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Random Polutation | Fitness score | Relative Fitness | Probability of selection | Probability distribution | Length |
| P5 | 6 -> 4 -> 3-> 2 -> 5 | 3162 | 18931/3162 = 5.987 | 0.237 | 0.237 | 5305 |
| C11 | 5 -> 4 -> 3-> 2 -> 6 | 3771 | 18931/3771 = 5.020 | 0.199 | 0.436 | 5753 |
| C12 | 5 -> 6 -> 3-> 2 -> 4 | 4278 | 18931/4278 = 4.425 | 0.175 | 0.611 | 6365 |
| C8 | 6 -> 4 -> 5-> 2 -> 3 | 3749 | 18931/3749 = 5.050 | 0.200 | 0.811 | 6385 |
| C10 | 2 -> 6 -> 3-> 4 -> 5 | 3971 | 18931/3971 = 4.767 | 0.189 | 1.00 | 6892 |
| Total Fitness: | | | 18931 |  |  |  |



A simple GA algorithm is described as follows

* Step1: Create a random population of potential solutions [24] consisting of n individuals (initial populations).
* Step2: Evaluate the fitness value f(x) of each individual, x, in the population.
* Step3: Repeat the following three steps to create a new population until completion of the new population.
* Step4: Select two individuals of the current generation for mating.
* Step5: Apply crossover with a certain ratio to create offspring.
* Step6: Apply mutation with a certain ratio.
* Step7: The previous operations are repeated until the completion criterion is met.

Thuật toán di truyền được mô tả chi tiết trong bài viết đang thực hiện giải bài toán người du lịch (TSP). Dưới đây là một phân tích từng bước:

Bước 1: Khởi tạo dân số ban đầu

Bước 2: Chọn bố và mẹ

Bố mẹ được chọn dựa trên điểm thích hợp của họ.

Sử dụng phân phối xác suất để chọn.

Số ngẫu nhiên được tạo ra để chọn bố mẹ.

Sử dụng crossover dựa trên xác suất crossover (0.8).

Bước 3: Sản xuất con

Chọn hai bố mẹ, và crossover được áp dụng để tạo ra hai con.

Crossover một điểm được sử dụng với một điểm crossover ngẫu nhiên.

Mutation được áp dụng để duy trì một giải pháp hợp lệ cho TSP.

Bước 4: Đột biến

Đột biến được áp dụng với một xác suất nhất định (0.1) cho mỗi con.

Số ngẫu nhiên được tạo ra để quyết định liệu có xảy ra đột biến hay không.

Bước 5: Thay thế

Tổng chiều dài của hành trình của dân số ban đầu và con cái được đánh giá.

Chọn các cá thể có hiệu suất tốt nhất để tạo ra thế hệ tiếp theo, với tỷ lệ bảo toàn là 10%.

Phép lặp:

Quá trình được lặp lại cho một số lần lặp xác định (4 trong trường hợp này).

Dân số mới được tạo ra, và các cá thể tốt nhất được chọn cho thế hệ tiếp theo.

Kết Quả Cuối Cùng

Sau nhiều lần lặp, thuật toán hội tụ thành một tập giải pháp, mỗi giải pháp đại diện cho một hành trình với điểm thích hợp đi kèm.

***Độ phức tạp của thuật toán:***

Thuật toán di truyền không hỗn loạn mà mang tính ngẫu nhiên. Độ phức tạp phụ thuộc vào các toán tử di truyền, việc thực hiện chúng (có thể có tác động rất đáng kể đến độ phức tạp tổng thể), sự đại diện của các cá thể và quần thể, và rõ ràng là vào chức năng thích nghi.

Đưa ra các lựa chọn thông thường (point mutation, one point crossover, roulette wheel selection) độ phức tạp của Thuật toán di truyền là O(g(nm + nm + n)) với g số thế hệ, n kích thước quần thể và m kích thước của các cá thể . Do đó độ phức tạp theo thứ tự O(gnm)).

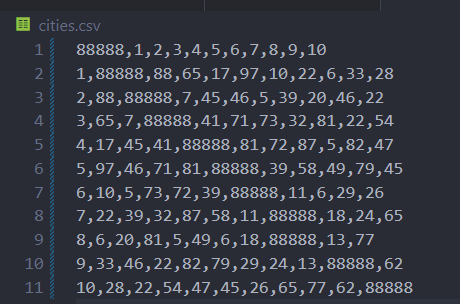
Nếu số lượng thế hệ và quy mô dân số không đổi, miễn là hàm đột biến, hàm chéo và hàm thích nghi của bạn mất một khoảng thời gian đã biết, thì o lớn là O(1) - nó cần một khoảng thời gian không đổi.

Bây giờ, nếu bạn hỏi chữ O lớn sẽ có ý nghĩa gì đối với dân số N và một số thế hệ M, thì điều đó lại khác, nhưng như đã nêu khi bạn biết trước tất cả các biến số, lượng thời gian thực hiện là không đổi đối với vào đầu vào của bạn.

* Độ phức tạp của thuật toán phụ thuộc vào các yếu tố như kích thước dân số, số lần lặp, và độ phức tạp của hàm fitness.
* Thuật toán bao gồm các bước như tạo số ngẫu nhiên, đánh giá fitness, chọn lọc, crossover, đột biến, và thay thế.
* Độ phức tạp thời gian tổng thể được ảnh hưởng bởi tích của kích thước dân số, số lần lặp và độ phức tạp thời gian của hàm fitness.
* Sự thành công của thuật toán phụ thuộc vào cách nó khám phá không gian giải pháp, cân bằng giữa khám phá và khai thác. Đa dạng của dân số, xác suất crossover và đột biến, và cơ chế chọn lọc đều ảnh hưởng đến hiệu suất của thuật toán. Tinh chỉnh các tham số này và thử nghiệm trên các trường hợp bài toán khác nhau có thể tối ưu hóa thuật toán cho TSP và các vấn đề tối ưu hóa kết hợp khác.
* Thuật toán di truyền có độ phức tạp thời gian là O(G \* P \* M), trong đó:
* G là số lần lặp (số thế hệ).
* P là kích thước dân số (population size).
* M là độ phức tạp của hàm fitness (evaluation function).

Do đó, độ phức tạp của thuật toán di truyền phụ thuộc vào số lần lặp, kích thước dân số và độ phức tạp của hàm fitness. Nếu chúng ta gia tăng bất kỳ yếu tố nào, độ phức tạp của thuật toán cũng sẽ tăng lên.

* 1. Viết chương trình tạo tập dữ liệu đầu vào



Hình 8: File data csv được generate

function generateTestData(filename):

    try

        create BufferedWriter writer for filename

        create Scanner scanner for user input

        output "Enter number of cities: "

        numCities = read integer from scanner

        // Write city indices to the file

        writer.write("88888,")

        for i from 1 to numCities:

            writer.write(String.valueOf(i))

            if i < numCities:

                writer.write(",")

        writer.newLine()

        create Random random

        // Create a square matrix for distances

        create 2D array distances with dimensions (numCities + 1) x (numCities + 1)

        for i from 1 to numCities:

            for j from 1 to numCities:

                distances[i][j] = -1

        // Write distances between cities to the file

        for i from 1 to numCities:

            writer.write(String.valueOf(i))

            for j from 1 to numCities:

                if distances[i][j] == -1:

                    if i == j:

                        distances[i][j] = 88888

                    else:

                        distances[i][j] = random.nextInt(100) + 1

                        distances[j][i] = distances[i][j]

                writer.write("," + distances[i][j])

            writer.newLine()

        close scanner

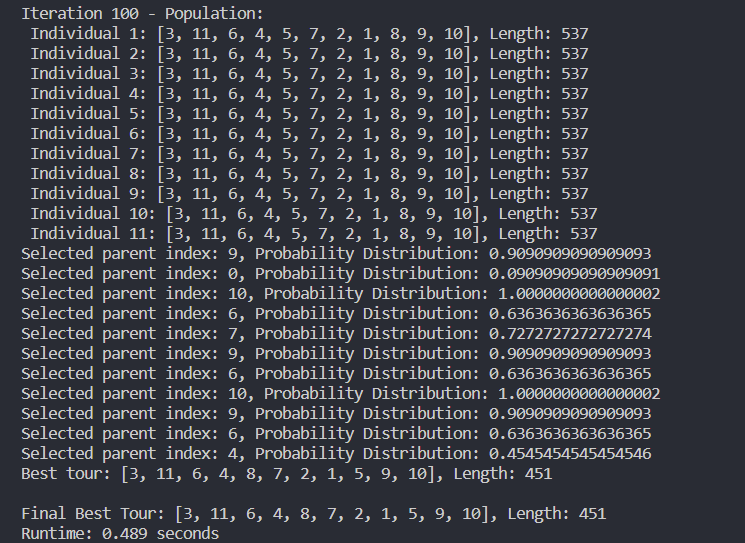
        close writer

    catch IOException e:

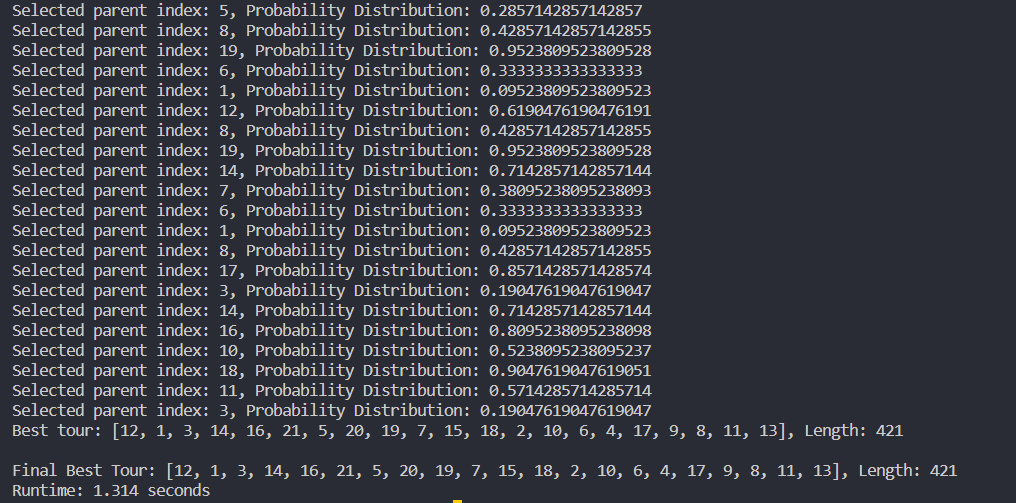
        // Handle IO errors (e.g., file not found, permission issues)

        print stack trace of e

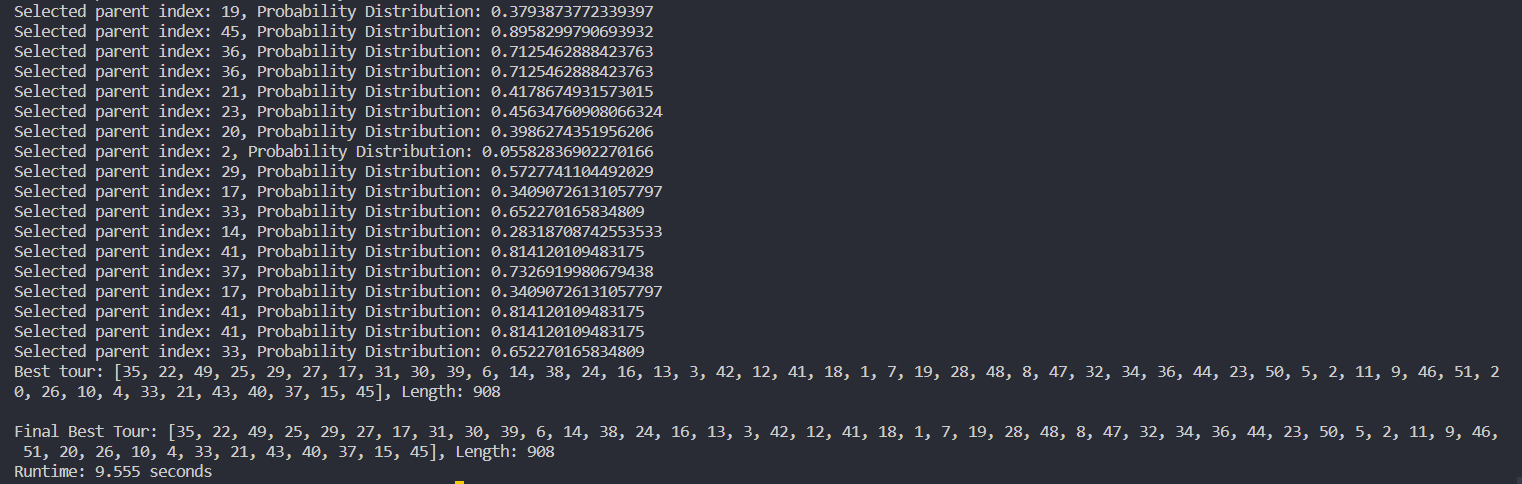
* 1. Thử nghiệm các dataset được generate ra



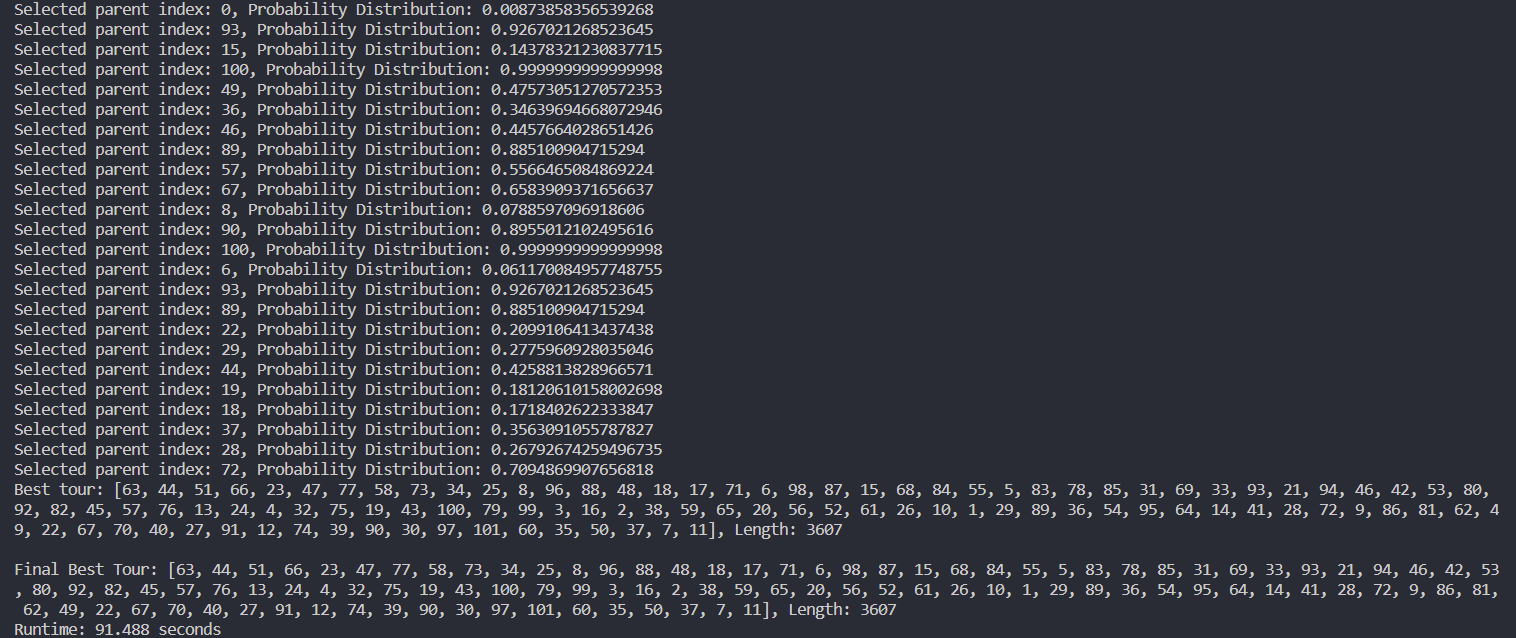
Hình 9: Kết quả thử nghiệm với 10 thành phố



Hình 10: Kết quả thử nghiệm với 20 thành phố

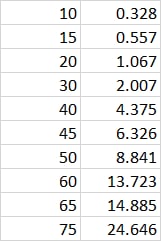
****

Hình 11: Kết quả thử nghiệm với 50 thành phố

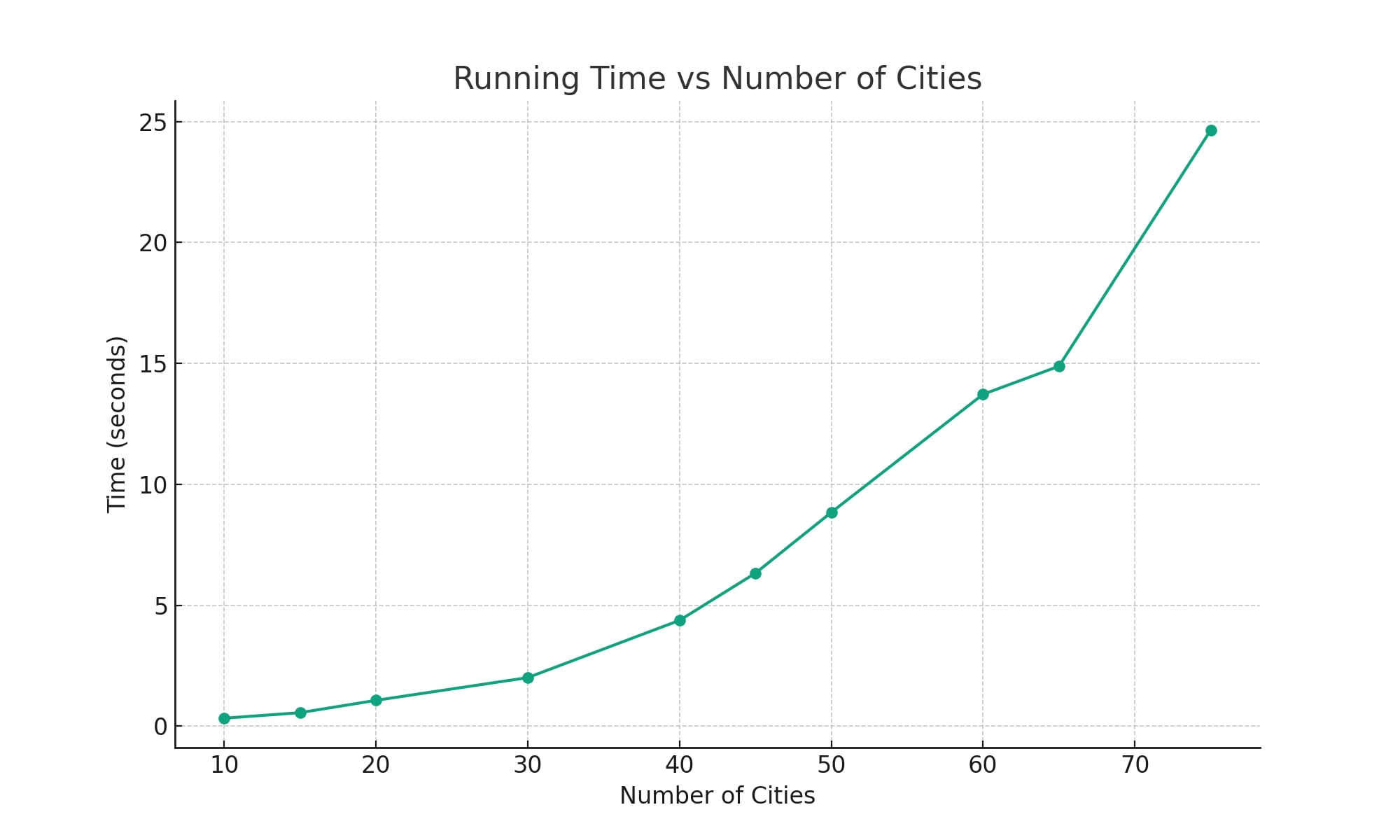
****

Hình 12: Kết quả thử nghiệm với 100 thành phố

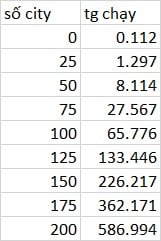
* 1. Vẽ biểu đồ thời gian và số thành phố chạy



Hình 13: Thời gian chạy của 75 thành phố



Hình 14: Biểu đồ thời gian chạy của 75 thành phố



Hình 15: Thời gian chạy của 200 thành phố



Hình 16: Biểu đồ thời gian chạy của 200 thành phố

* 1. So sánh so với thuật toán khác:

Bài báo đề xuất cách cải tiến thuật toán Ant Colony để hỗ trợ tìm ra đường đi ngắn hơn cho bài toán người bán hàng. Bài toán người bán hàng yêu cầu tìm ra đường đi ngắn nhất cho người bán hàng đi qua các thành phố và cuối cùng quay về lại thành phố xuất phát, mỗi thành phố chỉ được ghé thăm một lần, biết rằng tất cả các thành phố đều có đường đi đến với nhau và khoảng cách giữa các thành phố là biết trước. Có rất nhiều thuật toán giải quyết được bài toán này. Một trong những thuật toán được nghiên cứu nhiều và giải quyết khá hiệu quả cho bài toán này là thuật toán Ant Colony (thuật toán đàn kiến). Thuật toán Ant Colony có những hỗ trợ tìm kiếm rất mạnh mẽ và tỏ ra khá thích hợp với những bài toán có không gian tìm kiếm cực lớn. Tuy nhiên khi áp dụng thuật toán Ant Colony cho bài toán người bán hàng thì chi phí vẫn còn khá cao. Vì vậy nhóm chúng tôi thiết kế giải thuật cải tiến thuật toán Ant Colony để tìm lời giải tối ưu hơn cho bài toán người bán hàng. Chúng tôi đã tiến hành xây dựng thử nghiệm với nhiều bộ dữ liệu đầu vào để so sánh giữa thuật toán cải tiến với thuật toán Ant Colony nhằm đánh giá một cách chính xác và khách quan nhất. Kết quả cho thấy thuật toán cải tiến của nhóm chúng tôi đề xuất đã có cải thiện đáng kể về chi phí so với thuật toán Ant Colony.

Input: Số kiến, khởi tạo số thành phố, vị trí tọa độ các thành phố, điểm xuất phát, khởi tạo mùi.

Output: Chu trình đường đi ngắn nhất từ thành phố xuất phát qua các thành phố còn lại một lần và quay về lại thành phố ban đầu

Method:

Begin:

B1:Nhập dữ liệu đầu vào: số kiến, vị trí tọa độ các thành phố, khoảng cách giữa các thành phố, điểm xuất phát, khởi tạo mùi.

B2: Xây dựng phương án lựa chọn đường đi

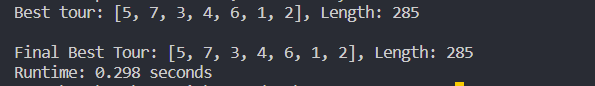
B3: Cập nhật mùi

B4: Nếu chu trình đường đi là ngắn nhất thì qua B5 ngược lại thì quay lại B2

End



Hình 17: TSP 6 thành phố chạy bằng thuật toán Ant Colony



Hình 18: TSP 6 thành phố chạy bằng thuật toán Genetic

TÀI LIỆU THAM KHẢO

**Tiếng Anh**

1. Garey MR, Johnson DS (1979) A guide to the theory of NP-completeness. W. H. Freeman & Co, New York.

2. Gouveia L, Leitner M, Ruthmair M (2017) Extended formulations and branch and-cut algorithms for the black-and-white traveling salesman problem. Eur J Oper Res 262(3):908–928.

3. Michail O, Spirakis PG (2016) Traveling salesman problems in temporal graphs. Theoret Comput Sci 634:1–23.

4. Rego C, Gamboa D, Glover F, Osterman C (2011) Traveling salesman problem heuristics: leading methods, implementations and latest advances. Eur J Oper Res 211(3):427–441.

5. Rocki K, Suda R (2013) High performance GPU accelerated local optimization in TSP. In: IPDPSW IEEE 27th international, pp 1788–1796.