# Ý tưởng

Ý tưởng của em áp dụng thuật toán GA để lai ghép các hành trình lịch sử giao hàng của target , lịch sử này được khởi tạo cẩn thận từ initsolution do đó kết quả xuất phát khá tiệm cận với kết quả tối đa của hàm mục tiêu

# Sơ đồ thuật toán chung

Sơ đồ thuật toán áp dụng GA cụ thể như sau

Chart, diagram, box and whisker chart

Description automatically generated

Hình 1 Sơ đồ thuật toán GA (cách 2)

Cụ thể các bước trên như sau

* Nhận các tham số của bài toán: Đọc từ file csv các thông số như ở bước mô hình hoá bài toán
* Tính khoảng cách giữa các khách hàng: Lập một ma trận khoảng cách giữa các khách hàng với nhau
* Kiểm tra điều kiện dừng: Nếu fitness không tăng sau 50 vòng lặp thì sẽ dừng

Các phần tiếp theo sẽ được giải thích kỹ ở các mục sau

## Mã hoá lời giải

Mã hoá lời giải: Mỗi lời giải gồm 2 trường sau đây:

* Lịch trình di chuyển drone: là lịch trình di chuyển của drone đi qua từng target và tại target đó giao bao nhiêu hàng. Cụ thể như sau
  + Ví dụ Drone\_1 = [[1,10],[2,20]],[[5,50][6,60]]
    - Phần màu đỏ là hành trình thứ nhất của drone 1 trong đó ghé thăm điểm khách hàng số 1 vào giao khối lượng 10, sau đó đến khách hàng 2 và giao khối lượng 20 rồi quay trở về kho
* Lịch sử nhận hàng của từng khách hàng: là lịch trình mà khách hàng đó nhận hàng của thiết bị nào và nhận trọng lượng là bao nhiêu. Cụ thể cách mã hoá được biểu diễn trong hình lịch sử nhận hàng bên dưới.
  + Ví dụ Customer\_1 = [[1,10], [2,10]]
    - Phần màu đỏ là ở lượt thứ nhất được thiết bị drone (id =1) giao khối lượng 10, sau đó đến drone (id = 2) giao khối lượng 10

Số liệu ở 2 trường đều đồng bộ với nhau, sở dĩ phải chia làm 2 trường để dễ dàng cho hàm kiểm soát giá trị rằng luôn thoả mãn ràng buộc của bài toán ở cả khách hàng và drone

## Init solution

Init solution gồm 3 cách khởi tạo:

### Init solution 1

Drone chọn điểm xuất phát là điểm gần nhất với nó, từ điểm đó di chuyển tới các điểm gần nhất và giao lower\_bound (nếu lower\_bound = 0 thì bỏ qua không cần giao) , dừng khi:

* Hết hàng để giao
* Hết thời gian duration của drone
* Hết thời gian working time
* Hết target

Sau đó tiếp tục đến xe tải, chọn điểm xa nhất làm điểm xuất phát rồi đi tới các điểm gần nhất cho đến khi đạt tới điều kiện dừng:

* Hết hàng để giao
* Hết thời gian working time
* Hết target

### Init solution 2

Drone chọn điểm xuất phát là điểm xa nhất với nó, từ điểm đó di chuyển tới các điểm gần nhất và giao lower\_bound (nếu lower\_bound = 0 thì bỏ qua không cần giao) , dừng khi:

* Hết hàng để giao
* Hết thời gian duration của drone
* Hết thời gian working time
* Hết target

Sau đó tiếp tục đến xe tải, chọn điểm gần nhất làm điểm xuất phát rồi đi tới các điểm gần nhất cho đến khi đạt tới điều kiện dừng:

* Hết hàng để giao
* Hết thời gian working time
* Hết target

### Init solution 3

Drone chọn điểm xuất phát là điểm bất kỳ, từ điểm đó di chuyển tới các điểm gần nhất và giao lower\_bound (nếu lower\_bound = 0 thì bỏ qua không cần giao) , dừng khi:

* Hết hàng để giao
* Hết thời gian duration của drone
* Hết thời gian working time
* Hết target

Sau đó tiếp tục đến xe tải cũng chọn điểm bất kỳ làm điểm xuất phát rồi đi tới các điểm gần nhất cho đến khi đạt tới điều kiện dừng:

* Hết hàng để giao
* Hết thời gian working time
* Hết target

Từ các init solution này em có thể tạo ra gần 100 init solution khác nhau trong khi triển khai. Kết quả của các init solution này cũng không tệ ( gần đạt feasible)

## Chọn lọc

### Độ thích nghi

Để tính độ thích nghi trước hết em dựa vào biến penalty để tính giá trị phạt, cụ thể hàm penalty em tính dưạ trên 3 thành phần chính:

* Phạt dựa trên việc giao cho target thiếu lower\_bound, nếu giao thiếu sẽ bị trừ trọng số nhân với tỷ lệ đã được chuẩn hoá theo trọng số max

Text

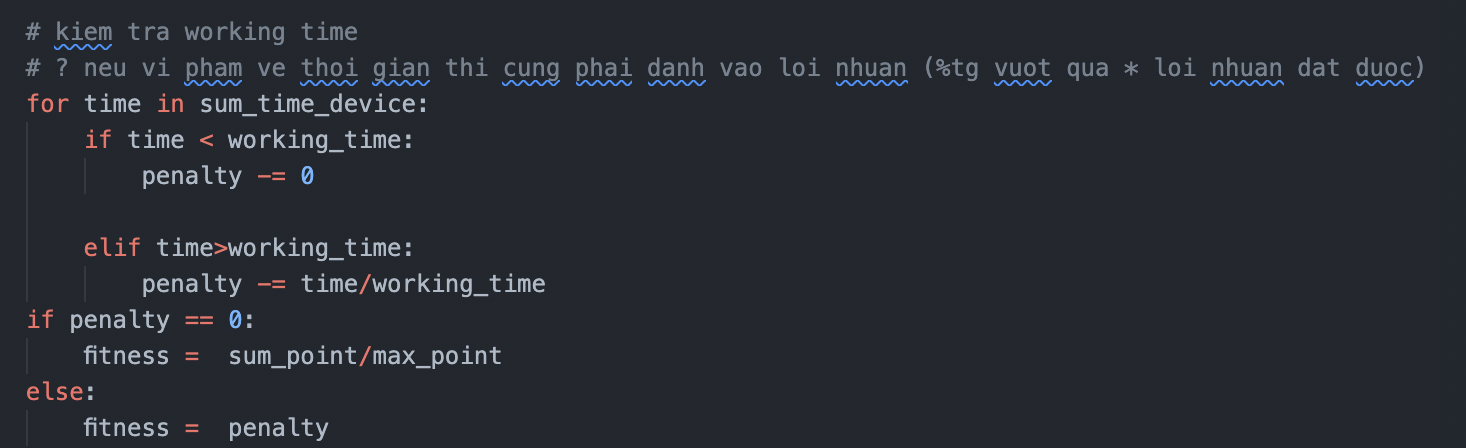
Description automatically generated

* Phạt dựa trên sự vi phạm của drone qua từng trip, nếu như trong 1 trip của drone mà vượt quá thời gian vi phạm cũng sẽ bị trừ điểm

Text

Description automatically generated

* Phạt dựa trên sự vi phạm working time của tất cả các thiết bị



Từ kết quả penalty tính ra em chia làm 2 trường hợp:

* Nếu penalty = 0 thì fitness sẽ bằng
  + Fitness = Giá trị chuẩn hoá hàm mục tiêu
* Nếu penalty < 0 thì fitness sẽ bằng
  + Fitness = penalty

### Chọn lọc

Dựa trên vòng quay roulette, chọn ngẫu nhiên các cá thể, các cá thể có thể xuất hiện nhiều lần trong quần thể mới.

* Đoạn code xây dựng vòng quay roulette như sau

Text

Description automatically generated

* Đoạn code lựa chọn ngẫu nhiên các cá thể trong vòng quay này

Graphical user interface, text, application

Description automatically generated

## Lai ghép

Chọn cá thể lai ghép có 2 cách, khi chọn có thể chọn ngẫu nhiên theo 2 cách sau:

* Dựa trên ma trận khác biệt
* Dựa trên chọn ngẫu nhiên

Lai ghép: Lai ghép giữa từng lịch sử nhận hàng của từng khách hàng ở cá thế này với từng lịch sử nhận hàng của từng khách hàng ở cá thể khác bằng cắch cắt từ một số ngâũ nhiên k ( điểm bắt đầu) và p( điểm kết thúc) và ném sang lịch sử nhận hàng phía bên kia cũng tại một số ngẫu nhiên p. Cụ thể như hình vẽ sau:

A picture containing schematic

Description automatically generated

Hình 2 Hình vẽ mô tả quá trình lai ghép

Ngoài ra trong khi triển khai em có đặt một tỷ lệ nhất định khi lai ghép chỉ cần chọn k, các điểm khác mặc định là điểm cuối cùng trong lịch sử để đảm bảo giữ được cấu trúc phân bố trong target nhiều nhất có thể

## Đột biến

Chọn ngẫu nhiên các cá thể để đột biến trong quần thể cũ, việc đột biến sẽ rơi vào một trong 3 trường hợp

* thực hiện đột biến khối lượng bất kỳ ở trong lịch sử giao hàng của từng khách hàng:
  + Ví dụ

|  |  |  |
| --- | --- | --- |
| Id target | Trước khi đột biến | Sau khi đột biến |
| 18 | [[0,25],[2,400]] | [[0,50],[2,400]] |

* Thực hiện đột biến tăng thêm trip mới cho target

|  |  |  |
| --- | --- | --- |
| Id target | Trước khi đột biến | Sau khi đột biến |
| 18 | [[0,25],[2,400]] | [[0,25],[2,400][3,10]] |

* Thực hiện đột biến xoá bớt trip cho target

|  |  |  |
| --- | --- | --- |
| Id target | Trước khi đột biến | Sau khi đột biến |
| 18 | [[0,25],[2,400]] | [[0,25]] |

## Education

Lựa chọn cá thể phải education: Chọn cá thể có fitness kém nhất

Hàm education học theo 2 hướng:

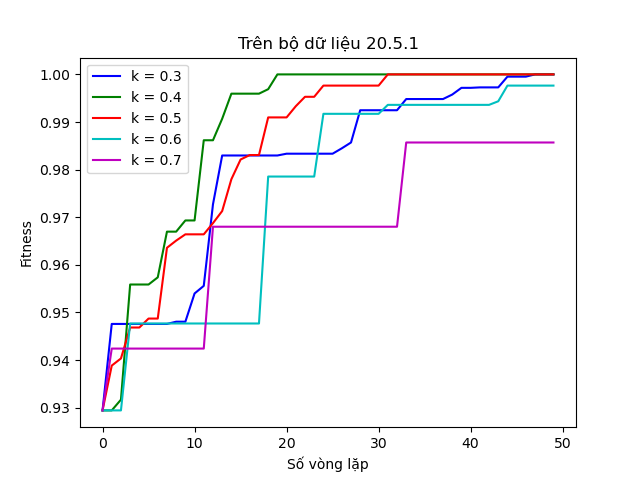
* Theo target: nếu target nào đang bị thiếu lowerbound thì sẽ được tăng khối lượng, nếu bị vượt quá upper\_bound sẽ được giảm khối lương
* Theo thiết bị: nếu target nào đang bị vượt quá giới hạn tải trọng thì tiến hành giảm khối lượng ở turn có số lượng hàng lớn nhất

# Lựa chọn tham số

Dưới đây là một vài tham số trong quá trình thực nghiệm em đúc rút ra được, ngoài ra còn một vài thông số khác như tỉ lệ lai ghép/ đột biến, tỉ lệ đột biến theo từng trường hợp được em tự điều chỉnh với từng bộ dữ liệu khác nhau cho kết quả tốt nhất.

## Lựa chọn tỷ lệ lai ghép ngẫu nhiên hoặc lai ghép theo trình tự

Trong quá trình làm em đã hiển thị kết quả fitness khi thay đổi tỉ lệ lai ghép ngẫu nhiên/lai ghép theo trình tự ở bộ dữ liệu 20.5.1. Kết quả như sau:



Hình 3 Biểu đồ thể hiện sự thay đổi độ dốc fitness với k (tỉ lệ lai ghép ngẫu nhiên/ lai ghép có trình tự) khác nhau

Từ biểu đồ trên có thể thấy ở k =0.4 tức là tỉ lệ lai ghép có trình tự nhiều hơn, đường fitness tăng rất nhanh. Từ phân tích cá nhân em cho rằng có 3 nguyên nhân sau:

* Nguyên nhân 1: do cấu trúc của init solution đã rất tốt ( ở phần khởi tạo đa phần đã đạt đến 93% so với kết quả tốt nhất) do vậy nếu trong quá trình lai ghép có thể giữ cấu trúc này thì sẽ giảm miền phải tìm kiếm đi dẫn đến đường k=0.4 hội tụ rất nhanh
* Nguyên nhân 2: khi k tăng lên thì miền tìm kiếm lời giải sẽ rất lớn, GA lao theo nhiều hướng tuy nhiên do giới hạn của quần thể dẫn tới việc có thể hướng đó là tốt, nhưng ở giai đoạn đầu khi chỉ lai ghép được 1 vài thành phần con dẫn tới nó chưa đủ tốt, fitness thấp hơn các cá thể khác dẫn đến bị loại. Và fitness liên tục đi ngang khi chưa kịp tìm ra hướng tốt hơn đã quên ( loại cá thể đó). Vấn đề này sẽ được em nghiên cứu tiếp ở phần sau mục Lựa chọn số lượng cá thể trong quần thể
* Nguyên nhân 3: Khi k tiếp tục giảm qua ngưỡng 0.4 thì kết quả đường fitness lại xấu đi, điều này là do với cấu trúc init solution ban đầu , việc lai ghép có trình tự có thể rất nhanh chóng đi tìm lời giải tốt nhất với cấu trúc đó, tuy nhiên với 30% dành cho sự ngẫu nhiên có thể là chưa đủ do cũng giống trường hợp 2, số lượng ngẫu nhiên ít dẫn tới những hướng tốt hơn nhưng kém về ban đầu rất khó cạnh tranh được với hướng có cấu trúc tốt như init solution.

Dựa vào phân tích nguyên nhân trên, em lựa chọn k = 0.4

## Lựa chọn số lượng cá thể trong quần thể

Như đã nói ở phần trên, việc chọn số lượng cá thể trong quần thể sẽ ảnh hưởng đến việc quần thể đó có thể nhớ đến mức nào, nếu số lượng cá thể ít thì với bộ dữ liệu nhỏ sẽ phù hợp, nhưng với bộ dữ liệu lớn thì sẽ dẫn tới giải thuật rất nhanh quên ( giai đoạn đầu xấu dù kết quả chung là tốt nhất vẫn sẽ bị loại). Cụ thể được thể hiện ở hình dưới.

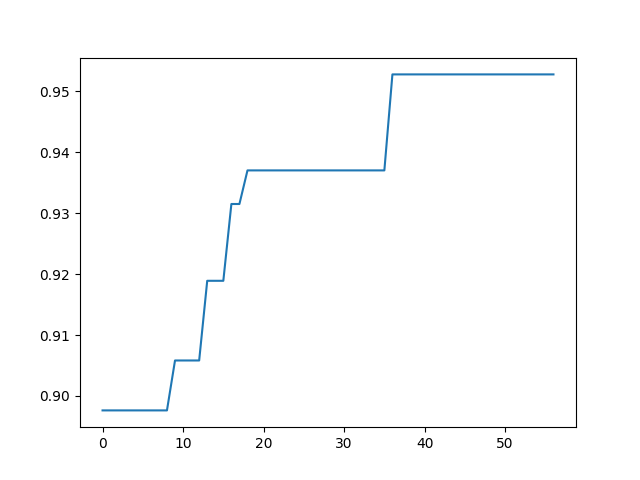
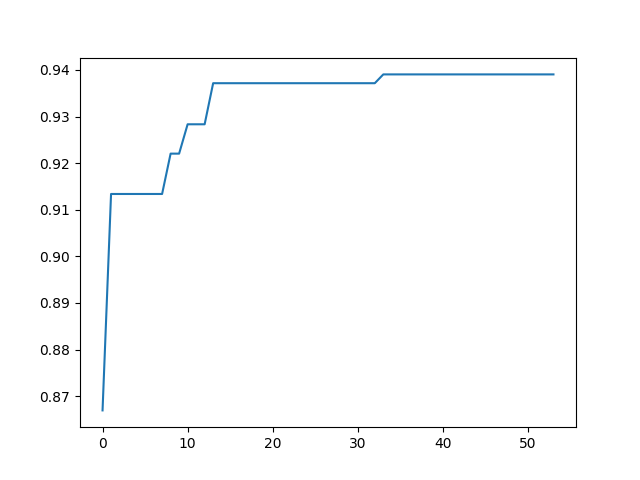
Chart, line chart

Description automatically generated

Hình 4 Đường fitness ở trên các bộ dữ liệu khác nhau với số lượng quần thể giữ nguyên

Từ đây có thể thấy đúng như dự đoán ban đầu, ở trên tập 6.5.1 với số lượng target nhỏ thì kết quả rất nhanh chóng hội tụ, nhưng càng về sau thì đường fitness càng đi ngang khi số target tăng lên do giới hạn của số lượng quần thể dẫn tới bị quên các cá thể chưa đủ tốt rất nhanh.

Từ đây em quyết định tăng số lượng quần thể và khi tăng số lượng cá thể trên bộ 20.10.1 em thu được kết quả như sau



Hình 5 Mối liên hệ giữa fitness và việc tăng quần thể theo hướng tăng lai ghép ( trái) và tăng đột biến (phải)

Kết quả trên cho thấy khi tăng số lượng quần thể theo hướng lai ghép ở ảnh bên trái thì kết quả tăng lên rất nhanh sau đó đi ngang do lại gặp giới hạn ở bên đột biến ( đột biến quên nhanh hơn) dẫn tới ở vòng lặp thứ 15 đổ đi đồ thị có xu hướng đi ngang

Khi tăng số lượng quần thể theo hướng đột biến thì sẽ liên tục lấy được kết quả tốt hơn ở những đột biến thành công tuy nhiên các đường đi ngang vẫn xuất hiện rải rác và càng về sau kéo càng dài do bên lai ghép quên nhanh hơn dẫn tới khó phát triển thêm từ những cấu trúc tốt do đột biến mang lại

Nhìn chung cả 2 hướng khi tăng số lượng quần thể lên 181 đều cho kết quả tốt hơn ở số lượng quần thể là 141. Củng cố thêm sự phân tích của em khá chính xác. Cùng với sự phân tích trên em quyết định lựa chọn khi tăng số lượng quần thể sẽ tăng đều cả lai ghép và đột biến

# Kết quả thu được

## Kết qủa tổng hợp

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Data Name | working time | number truck | number drone | Feasible | best solution | số thế hệ |
| 6.5.1 | 60 | 2 | 2 | Có | 2100 | 1 |
| 10.5.1 | 60 | 2 | 2 | Có | 28700 | 3 |
| 20.5.1 | 60 | 2 | 2 | Có | 53100 | 25 |
| 20.10.1 | 60 | 2 | 2 | Có | 56725 | 45 |
| 20.10.2 | 60 | 2 | 2 | Có | 30145 | 34 |
| 50.10.1 | 60 | 4 | 5 | Có | 95750 | 100 |
| 100.10.1 | 60 | 8 | 6 | Có | 165650 | 124 |
| 200.10.1 | 60 | 20 | 30 | Có | 396250 | 134 |

## Kết quả chi tiết

Kết quả trên bộ 6.5.1

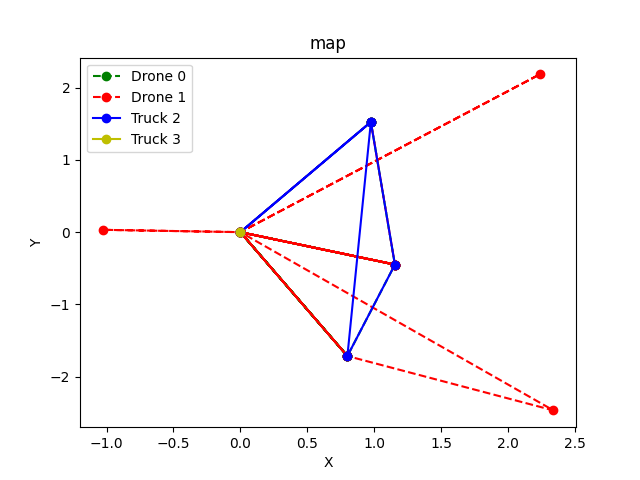
Độ tăng fitness được biểu thị ở đồ thị dưới

Chart, line chart

Description automatically generated

Hình Đồ thị fitness qua từng vòng lặp

Dưới đây là kết quả sau quá trình tính toán ra giá trị tối ưu



Hình Bản đồ giao hàng của drone và truck sau khi lập lịch

Kết quả trên bộ 20.10.1

Chart, box and whisker chart

Description automatically generated

Hình Biểu đồ sự tăng fitness qua từng vòng lặp

Hành trình tính ra như sau

Chart, radar chart

Description automatically generated

Hình Bản đồ giao hàng Drone và Truck được tính ra

Bộ dữ liệu 20.10.2

Chart

Description automatically generated

Cụ thể bản đồ hành trình drone và truck như sau

Chart, line chart

Description automatically generated

# Kết luận

Từ quá trình nghiên cứu em rút ra được một số điểm mạnh và điểm yếu của phần triển khai GA của em như sau:

* Điểm mạnh:
  + Rất nhanh chóng đạt được kết quả feasible
  + Có nhiều lựa chọn để lai ghép, đột biến cả theo hướng ngẫu nhiên và cấu trúc giúp mô hình có thể đạt được trạng thái tốt hơn
* Điểm yếu:
  + Phụ thuộc khá nhiều vào độ tốt của init solution, nếu init solution không đủ tốt thì phải mất thời gian rất lâu để mô hình có thể cải thiện về mặt kết quả
  + Chưa giữ được những bộ khung của lịch sử của device tốt cho vòng lặp tiếp theo do chưa có sự lai ghép ở phần device
  + Nhiều tham số cần phải điều chỉnh cho phù hợp với từng bài toán

Từ những đúc rút kinh nghiệm trên trong tương lai em sẽ tiếp tục cải thiện lời giải, trước mắt là cải thiện phần lai ghép để giữ được khung tốt của thiết bị. Những cập nhập đó sẽ được thể hiện ở link github: <https://github.com/congdinhchi/scheduled-delivery>.