**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



BÁO CÁO MÔN HỌC

**Tối ưu và lập kế hoạch**

***Đề tài:* Phân luồng cách ly COVID 19**

**Giảng viên hướng dẫn: Bùi Quốc Trung**

**Mã lớp học: 136049**

**Danh sách sinh viên thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Lớp** |
|  |  |  |  |
| 1 | Nguyễn Thành Phong | 20192016 | CTTN-KHMT-K64 |
|  |  |  |  |
| 2 | Trịnh Hồng Phượng | 20190062 | CTTN-KHMT-K64 |
|  |  |  |  |
| 3 | Nghiêm Việt Thắng | 20190088 | CTTN-KHMT-K64 |
|  |  |  |  |
| 4 | Hữu Tường Tú | 20194395 | CTTN-KHMT-K64 |
|  |  |  |  |

**MỤC LỤC**

[I, Giới thiệu đề tài 4](#_Toc1931183091)

[II, Tóm tắt bài toán 5](#_Toc49561160)

[1, Kí hiệu 5](#_Toc1117640651)

[2, Biến 5](#_Toc1550326995)

[3, Điều kiện 5](#_Toc319114811)

[4, Hạm mục tiêu 5](#_Toc55380042)

[III, Các giải thuật 6](#_Toc1791006166)

[1, Local search (Heuristic) 6](#_Toc1632001041)

[1.1, Phương pháp: 6](#_Toc1050687484)

[1.2, Các vấn đề: 6](#_Toc894372646)

[2, Tabu search (MetaHeuristic) 6](#_Toc1776056579)

[IV, Giải thuật tối ưu 7](#_Toc1594387063)

[V, Đánh giá 8](#_Toc329086651)

# **I, Giới thiệu đề tài**

Có N đoàn du khách 1,2,3,…,N từ các sân bay cần được đưa về các khu cách ly 1,2,…,M trong bối cảnh COVID19. Đoàn khách i có s(i) người. Khu lách ly j có thể chưa được c(j) người (số chỗ là c(j))

Đoàn khách thứ i nằm ở điểm i (i=1,…,N) và khu cách ly j nằm ở điểm N+j (j=1,…,M)

Hãy xây dựng phương án cách ly sao cho:

1. Mỗi đoàn khách đưa về đúng 1 khu cách ly

2. Tổng số người đưa về mỗi khu cách ly không vượt quá số chỗ của khu cách ly đó.

3. Khoảng cách lớn nhất từ điểm xuất phát của đoàn khách nào đó đến khu cách ly được phân công là ngắn nhất

# **II, Tóm tắt bài toán**

## 1, Kí hiệu

N : số đoàn khách

M : số khu cách li

S[i] : thông tin của đoàn khách thứ i

C[i] : thông tin của khu cách ly thứ i

( các thông tin này gồm có :

location : tọa độ chiều của đối tượng

count : số lượng thành viên (với đoàn khách) hoặc số lượng chứa tối đa của khu cách ly)

Distance( a, b): Khoảng cách ơ-clit từ đối tượng a đến đối tượng b

## 2, Biến

X[i] : Đoàn cách li thứ i sẽ di chuyển tới khu cách ly X[i],

## 3, Điều kiện

với mọi

## 4, Hàm mục tiêu

Ta định nghĩa thêm maxdistance\_X là vị trị i của đoàn cách li phải di chuyển ra xa nhất( tức có ) của 1 lời giải X.

# **III, Các giải thuật**

## 1, Local search (Heuristic)

### 1.1, Phương pháp:

Khởi tạo một lời giải trong không gian tìm kiếm, từ đó đi tới các lời giải có giá trị hàm mục tiêu tốt hơn.

### 1.2, Thực hiện:

Lân cận của một lời giải hiện tại có rất nhiều cách định nghĩa một lân cận cho lời giải. Dựa vào bài toán đang xét nhóm định nghĩa một lân cận Y của 1 lời giải X nếu thỏa mãn 1 trong 2 điều kiện:

Điều kiện 1:

Y[ maxdistance\_X] # X[maxdistance\_X]

Y[ i] = X[i] với mọi 0<= i<= N-1, I # maxdistance\_X

Điều kiện 2:

Tồn tại vị trí replace\_X # maxdistance\_X và ta sẽ đổi vị trí di chuyển của 2 đoàn khách này, cụ thể :

Y[ replace\_X ] = X[ indexdistance\_X]

Y[ indexdistance\_X] = Y[ replace\_X]

Ví dụ:

N = 5, M = 8( maxindex\_X = 2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| I | 0 | 1 | 2 | 3 | 4 |
| X[i] | 5 | 0 | 2 | 4 | 6 |
| Y[i] ( điều kiện 1) | 5 | 0 | 3 | 4 | 6 |
| Y[i] ( điều kiện 2, replace\_X= 1 ) | 5 | 2 | 0 | 4 | 6 |

1.3 Ưu điểm:

Vấn đề trạng thái ‘shoulder’ và điểm ‘flat’ local minimize không xảy ra do hàng xóm mới không thể có hàm mục tiêu trùng với ban đầu.

Triển khai dễ dàng

1.4 Nhược điểm:

Không gian tìm kiểm khá hẹp, dẫn tới hội tụ sớm.

## 2, Tabu search (MetaHeuristic)

2.1. Phương pháp:

Phương pháp Tabu Search bắt đầu từ một lời giải ngẫu nhiên của bài toán, sau đó sẽ áp dụng một số các toán tử lân cận để sinh ra lời giải mới, đồng thời lưu lại các lời giải dã đi qua với mục đích thoát khỏi local-minimize, quá trình này được lặp đi lặp lại cho đến khi thỏa mãn điều kiện kết thúc của thuật toán.

2.2. Thực hiện:

Dựa vào bài toán đang xét nhóm định nghĩa một lân cận Y của 1 lời giải X nếu thỏa mãn 1 trong 2 điều kiện:

Điều kiện 1:

Tồn tại 1 vị trí replace\_X sao cho

Y[ replace\_X] # X[replace\_X]

Y[ i] = X[i] với mọi 0<= i<= N-1, I # replace\_X

Điều kiện 2:

Tồn tại 2 vị trí i và j và đổi vị trí cách ly tại 2 vị trí này trong X để được Y, cụ thể:

Y[i] = X[j]

X[j] = Y[j]

Y[ k] = X[k], 0<= k<= N-1, k # i,j

Việc lưu trữ các lời giải của bài toán dẫn tới việc bộ nhớ bị tăng lên một cách nhanh chóng, đồng thời việc kiểm tra hàng xóm có ở trong kho lưu trữ không cũng khó khăn. Do vậy số lượng lời giải khi lưu trữ thường giữ ở mức nhất định tùy vào độ lớn của dữ liệu đầu vào.

2.3 Ưu điểm:

Có thể thoát khỏi local-minimize

Cho kết quả thường khá tốt

2.4 Nhược điểm:

Thời gian chạy lớn, bộ nhớ sử dụng lớn

Có thể rơi vào vòng lặp khi có 2 vùng local-minimize chuyển qua lại

# **IV, Giải thuật tối ưu**

4.1 Thuật toán Or-tool

4.1.1 Phương pháp

Thuật toán ortool được phát triển bới Google giúp cho việc giải các bài toán CP trở nên dễ dàng. Với lớp bài toán CP,thuật toán or-tool tỏ ra hiệu trong việc giải quyết để tìm ra nghiệm tối ưu toàn cục thay vì các thuật toán heuristic, tuy vậy về mặt thời gian chạy không tính được trước dẫn tới cần thực nghiệm trên nhiều bộ dữ liệu khác nhau.

4.1.2 Triển khai

Với cách mô hình hóa ban đầu, hiện tại chưa tìm ra cách để cài đặt trong các ‘constrain’ của or-tool nên nhóm đề xuất sử dụng mảng 2 chiều để biểu diễn lời giải.

Các ràng buộc về biến của bài toán biểu diễn được dưới các mẫu ‘constrain’ sẵn có, tuy vậy hàm mục tiêu với hàm max không được hỗ trợ. Vì vậy nhóm đề xuất 2 phương án giải quyết.

Phương án 1:

Dựa vào hàm mục tiêu yêu cầu giảm thiểu khoảng cách di chuyển lớn nhất của một đoàn khách, do vậy ta có thể dùng phương pháp chặt nhị phân để tìm kết quả tối ưu. Cụ thể, nếu tìm được một lời giải X đưa ra giá trị hàm mục tiêu là f(X) thì những lời giải có giá trị hàm mục tiêu lớn hơn hoặc bằng X có thể bỏ qua. ‘Constrain’ có thể được viết lại như sau:

Biến:

X[i][j] : đoàn khách i di chuyển tới khu cách ly j thì X[i][j] = 1, ngược lại X[i][j] = 0 với 0<= i<= N-1, 0<= j<= M-1

H : giá trị hàm mục tiêu ước lượng.

Điều kiện:

, H

Hàm mục tiêu:

Tồn tại lời giải của bài toán

Phương án 2:

Với hàm max không có ‘constrain’ có sẵn, nhưng hoàn toàn có thể biểu diễn được dưới các constrain khác và biến mới.

Biến:

X[i][j] : đoàn khách i di chuyển tới khu cách ly j thì X[i][j] = 1, ngược lại X[i][j] = 0 với 0<= i<= N-1, 0<= j<= M-1

Value[i] : biến giả kiểu int, 0<= i<= N-1

Bool\_[i] : biến giả kiểu boolean, 0<= i<= N-1

Điều kiện:

Hàm mục tiêu:

F = min( )

4.2 Đồ thị

Sau khi tìm hiểu, nhóm nhận thấy rằng bài toán không nằm trong lớp bài toán NP khó. Giống như phương án 1 của thuật ortool, dựa vào phương pháp chặt nhị phân để tìm kiếm lời giải bài toán, sau đó xây dựng đồ thị 2 phía như sau với giá trị hàm mục tiêu ước lượng là H:

Tập hợp A gồm có N đỉnh đại diện cho N khu cách ly và tập hợp B có M đỉnh đại diện cho M khu cách ly.

1 Đỉnh a ở tập hợp A được nối với một đỉnh b ở tập hợp B nếu khoảng cách ơ-clit từ đoàn khách a tới khu cách ly b bé hơn hoặc bằng H.

Mỗi cách nối N đỉnh khác biệt của tập hợp A với N đỉnh khác biệt của tập hợp B tương ứng với một lời giải của bài toán với giá trị hàm mục tiêu ước lượng là H.

* Sau đó sử dụng thuật toán ghép cặp cực đại để tìm số cặp ghép tối đa, nếu như có đủ N cặp ghép tức tồn tại lời giải cho giá trị H, và ngược lại.

Độ phức tạp của giải thuật : O( N + M) ^ 3 \* log( M\*N) )

V, Dữ liệu:

Dữ liệu được tạo ra bằng cách sinh ngẫu nhiên input của bài toán và sinh input trong các trường hợp ngoại lệ mà sinh ngẫu nhiên tí tạo ra tùy thuộc thuật toán được áp dụng giải bài toán.

1. Dữ liệu sinh ngẫu nhiên:

2 tập dữ liệu được sinh ngẫu nhiên:

Dữ liệu 1:

N = 10, M = 20, 50<= S[i].count <= 100, 25<= C[i].count<= 125

-100 <=S[i].x,S[i].y,C[i].x,C[i].y <= 100

Dữ liệu 2:

N = 15, M = 32, 100<= S[i].count <= 200, 70<= C[i].count<= 300

-1000 <=S[i].x,S[i].y,C[i].x,C[i].y <= 1000

1. Dữ liệu tự sinh

Tùy vào thuật toán, dữ liệu được tạo ra trong các trường hợp đặc biệt, thường khiến tốc độ của thuật toán rơi vào trường hợp xấu. Nhóm đã sinh các trường hợp đặc biệt, chủ yếu rơi vào các lỗi mà khi viết mã không nhận ra và đã sửa theo những lỗi đó. Những trường hợp gây tốn thời gian do thuật toán không xác định được độ phức tạp nên không có data cho trường hợp này.

# **VI, Đánh giá kết quả**

6.1 Trên bộ dữ liệu tự sinh:

Dữ liệu 1 chạy trên 100 bộ dữ liệu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Thuật toán | Local-search | Tabu-search | Ortool-pp1 | Ortool-pp2 | Đồ thị |
| Tỉ lệ so với giá trị tối ưu | 169% | 109% | 100% | 100% | 100% |
| Tổng thời gian chạy trên 100 bộ (giây) | 0.45 | 64 | 18.8 | 14.7 | 0.17 |

Dữ liệu 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Thuật toán | Local-search | Tabu-search | Ortool-pp1 | Ortool-pp2 | Đồ thị |
| Tỉ lệ so với giá trị tối ưu | 279% | 108% | 100% | 100% | 100% |
| Thời gian chạy( giây) | 0.87 | 261.24 | 39.1 | 42.3 | 0.46 |

VII, Tài liệu tham khảo

https://sites.google.com/site/kc97ble/algorithm-graph/cap-ghep-tren-do-thi-hai-phia-khong-trong-so