**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP THỰC PHẨM TP.HCM**



**ĐỒ ÁN:**

**XẾP LỊCH TRỰC CHO BỆNH VIỆN ĐA KHOA**

***GVHD: Ngô Dương Hà***

TP.HCM, ngày 30 tháng 07 năm 2020

**🙡🙞❖🙜🙣**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và Tên** | **MSSV** | **Công việc** |
| 1 | Nguyễn Minh Hiến | 2001180169 | Tìm hiểu về giải thuật di truyền(phần giao đấu, lai ghép)+tổng hợp ý kiến+ Chỉnh sửa +word |
| 2 | Trần Đình Văn | 2001180274 | Hỗ trợ + Tìm kiếm tài liệu tham khảo+ Form+ power point |
| 3 | Lương Viễn Đạt |  | Tìm hiểu về giải thuật di truyền(Đột biến, độ thích nghi)+ Hỗ trợ |

BẢNG PHÂN CÔNG CÔNG VIỆC NHÓM 4

**MỤC LỤC**

[BẢNG PHÂN CÔNG CÔNG VIỆC NHÓM 4 2](#_Toc47300564)

[LỜI NÓI ĐẦU 4](#_Toc47300565)

[I. LÝ THUYẾT VỀ GIẢI THUẬT DI TRUYỀN (GENETIC ALGORITHM - GA) 5](#_Toc47300566)

[**1. Lịch sử của giải thuật di truyền 5**](#_Toc47300567)

[II. BIỂU DIỄN BÀI TOÁN TRONG GIẢI THUẬT DI TRUYỀN 5](#_Toc47300568)

[**1. Giải thuật di truyền 5**](#_Toc47300569)

[**1.1. Độ thích nghi tiêu chuẩn 8**](#_Toc47300570)

[**Hàm mục tiêu là hàm dùng để đánh giá độ tốt của cá thể. Hàm mục tiêu 8**](#_Toc47300571)

[**trị của số thực này mà ta biết độ tốt của cá thể đó. 8**](#_Toc47300572)

[**1.2. Các phương pháp chọn lọc 8**](#_Toc47300573)

[**1.3. Các phương pháp lai ghép 8**](#_Toc47300574)

[**1.4. Các phương pháp đột biến 8**](#_Toc47300575)

[II. BÀI TOÁN SẮP XẾP LỊCH TRỰC BỆNH VIỆN ĐA KHOA 9](#_Toc47300576)

[**1. Giới thiệu bài toán 9**](#_Toc47300577)

[**2. Dữ liệu bài toán 9**](#_Toc47300578)

[**3. Các ràng buộc của bài toán 9**](#_Toc47300579)

[**4. Áp dụng thuật toán giải thuật di truyền vào bài toán 10**](#_Toc47300580)

[**4.1. Khởi tạo các class cho bài toán 10**](#_Toc47300581)

[**4.2. Khởi tạo quần thể 10**](#_Toc47300582)

[**4.3. Lai ghép và đột biến 11**](#_Toc47300583)

[**5. Chọn điểm dừng thuật toán 12**](#_Toc47300584)

[III. KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN 13](#_Toc47300585)

[**1. Kết luận. 13**](#_Toc47300586)

[**2. Hạn chế 13**](#_Toc47300587)

[**3. Hướng phát triển. 13**](#_Toc47300588)

[IV. TÀI LIỆU THAM KHẢO 14](#_Toc47300589)

LỜI NÓI ĐẦU

Với khả năng hiện nay, máy tính đã giúp con người giải quyết được rất nhiều bài toán khó mà trước kia thường bó tay. Mặc dù vậy vẫn còn một số lớn các bài toán thú vị nhưng chưa có thuật giải hợp lý để giải chúng. Trong số đó các bài toán tối ưu thường gặp trong thực tiễn.

Trước kia để giải những bài toàn tối ưu người ta thường dùng những phương pháp cổ điển như: leo đồi, mô phỏng luyện kim… Với những bài toán có không gian tìm kiếm nhỏ. Thì những phương pháp trên có thể giải quyết tốt. Nhưng với không gian tìm kiếm lớn, thì những phương pháp trên không hiệu quả. Vì vậy, điều kiện đòi hỏi chúng ta phải có những phương pháp mới để có thể giải quyết tốt những bài toán dạng trên. Ngày nay để giải bài toán tối ưu, chúng ta có thể dùng ”giải thuật di truyền” .

I. LÝ THUYẾT VỀ GIẢI THUẬT DI TRUYỀN (GENETIC ALGORITHM - GA)

1. Lịch sử của giải thuật di truyền

Giải thuật di truyền được phát triển dựa trên sự mô phỏng quá trình tiến hóa của sinh học. Được bắt đầu bằng Nils Aall Baricelli mô phỏng quá trình tiến hóa trong trò chơi năm 1945. Sau đó đến Alex Fraser xuất bản cuối sách Artificial Selection (Chọn lọc nhân tạo). Nhưng John Holland mới là người đầu tiên thực sự đặt tên cho giải thuật là “giải thuật di truyền” bằng việc xuất bản cuốn sách năm 1975. Từ đây giải thuật đã có tên là “giải thuật di truyền”. Và cùng với đó là sự phát triển mạnh mẽ hoàn thiện lý thuyết “giải thuật di truyền”. Và ứng dụng của giải thuật trong các bài toán thực tế.

Trước tiên ý niệm về thuật giải di truyền đã được một số nhà sinh vật học đưa ra từ những năm 50-60, thế kỷ XX. Alex Fraser là người tiên phong nêu lên sự tương đồng giữa sự tiến hóa của sinh vật và chương trình tin học giả tưởng về genetic algorithm. Tuy nhiên, John Henry Holland mới là người triển khai ý tưởng và phương thức giải quyết vấn đề dựa theo sự tiến hóa của con người. Từ những bài giảng, bài báo của mình, ông đã đúc kết các ý tưởng vào trong cuốn sách đầu tay *Adaptation in Natural and Artifical systems (mô phỏng theo tự nhiên và hệ thống nhân tạo )*, xuất bản năm 1975. Dựa trên lý thuyết cơ bản về GA của Holland, Keneth De Jong đã triển khai, chứng minh và những thành quả do ông thực hiện đã góp phần quan trọng trong việc tạo ra nền tảng toán học cho lý thuyết thuật giải di truyền. Và sau này là John koza đã tiếp nối, phát triển giải thuật di truyền.

Lần đầu tiên Holland nghiên cứu các thuật giải này, chúng hoàn toàn không có tên. Do nguồn gốc của phương pháp này là từ các gen di truyền, Holland đã đặt tên cho nó là “thuật giải di truyền“.

II. BIỂU DIỄN BÀI TOÁN TRONG GIẢI THUẬT DI TRUYỀN

1. Giải thuật di truyền

Thuật giải di truyền (GA) là kỹ thuật chung giúp giải quyết vấn đề bài toán bằng cách mô phỏng sự tiến hóa của con người hay của sinh vật nói chung (dựa trên thuyết tiến hóa muôn loài của Darwin) trong điều kiện qui định sẵn của môi trường. GA là một thuật giải, nghĩa là mục tiêu của GA không nhằm đưa ra lời giải chính xác tối ưu mà là đưa ra lời giải tương đối tối ưu.

Theo đề xuất ban đầu của giáo sư John Holland, một vấn đề-bài toán đặt ra sẽ được mã hóa thành các chuỗi bit với chiều dài cố định. Nói một cách chính xác là các thông số của bài toán sẽ được chuyển đổi và biểu diễn lại dưới dạng các chuỗi nhị phân. Các thông số này có thể là các biến của một hàm hoặc hệ số của một biểu thức toán học. Người ta gọi các chuỗi bit này là mã *genome* ứng với mỗi cá thể, các genome đều có cùng chiều dài. Nói ngắn gọn, một lời giải sẽ được biểu diễn bằng một chuỗi bit, cũng giống như mỗi cá thể đều được quy định bằng gen của cá thể đó vậy. Như vậy, đối với thuật giải di truyền, một cá thể chỉ có một gen duy nhất và một gen cũng chỉ phục vụ cho một cá thể duy nhất.

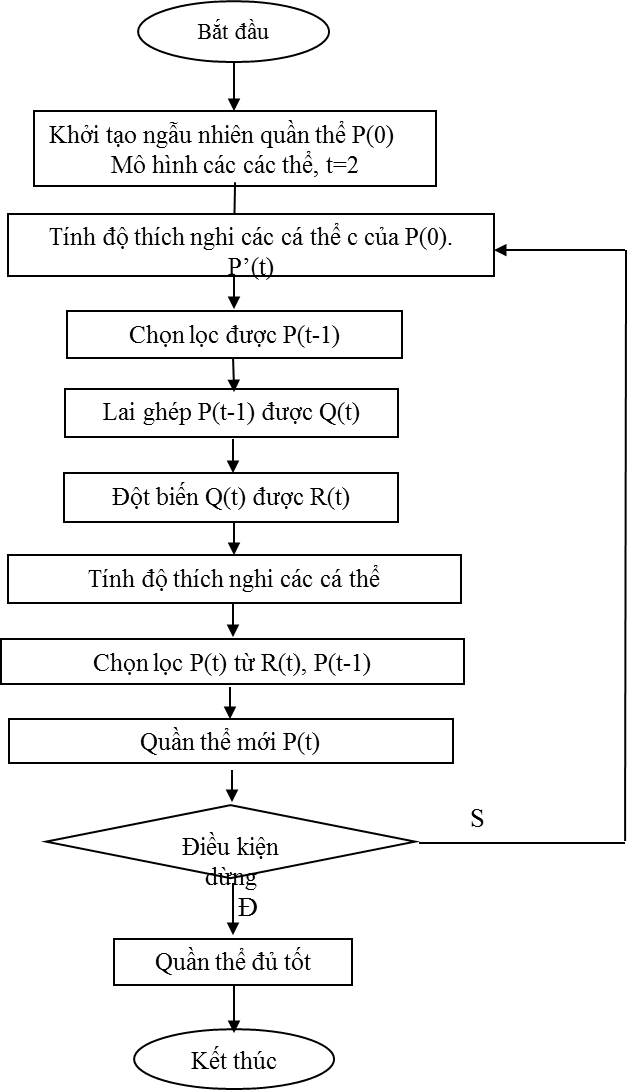
Ban đầu, ta sẽ phát sinh một số lượng lớn, giới hạn các cá thể có gen ngẫu nhiên. Nghĩa là phát sinh một tập hợp các chuỗi bit ngẫu nhiên. Tập các cá thể này được gọi là quần thể ban đầu (initial population). Sau đó, dựa trên một hàm nào đó, ta sẽ xác định được một giá trị gọi là độ thích nghi - Fitness. Giá trị này, có thể hiểu chính là độ "tốt" của lời giải. Vì phát sinh ngẫu nhiên nên độ "tốt" của lời giải hay tính thích nghi của các cá thể trong quần thể ban đầu là không xác định.

Để cải thiện tính thích nghi của quần thể, người ta tìm cách tạo ra quần thể mới. Có hai thao tác thực hiện trên thế hệ hiện tại để tạo ra một thế hệ khác với độ thích nghi tốt hơn. Thao tác đầu tiên là sao chép nguyên mẫu một nhóm các cá thể tốt từ thế hệ trước rồi đưa sang thế hệ sau (selection). Thao tác này đảm bảo độ thích nghi của thế hệ sau luôn được giữ ở một mức độ hợp lý. Các cá thể được chọn thông thường là các cá thể có độ thích nghi cao nhất.

Thao tác thứ hai là tạo các cá thể mới bằng cách thực hiện các thao tác *sinh sản* trên một số cá thể được chọn từ thế hệ trước – thông thường cũng là những cá thể có độ thích nghi cao. Có hai loại thao tác sinh sản : một là lai tạo (crossover), hai là đột biến (mutation). Trong thao tác lai tạo, từ gen của hai cá thể được chọn trong thế hệ trước sẽ được phối hợp với nhau (theo một số quy tắc nào đó) để tạo thành hai gen mới.

Thao tác chọn lọc và lai tạo giúp tạo ra thế hệ sau. Tuy nhiên, nhiều khi do thế hệ khởi tạo ban đầu có đặc tính chưa *phong phú* và chưa phù hợp nên các cá thể không rải đều được hết không gian của bài toán . Từ đó, khó có thể tìm ra lời giải tối ưu cho bài toán. Thao tác đột biến sẽ giúp giải quyết được vấn đề này. Đó là sự biến đổi ngẫu nhiên một hoặc nhiều thành phần gen của một cá thể ở thế hệ trước tạo ra một cá thể hoàn toàn mới ở thế thệ sau. Nhưng thao tác này chỉ được phép xảy ra với tần suất rất thấp (thường dưới 0.01), vì thao tác này có thể gây xáo trộn và làm mất đi những cá thể đã chọn lọc và lai tạo có tính thích nghi cao, dẫn đến thuật toán không còn hiệu quả.

Thế hệ mới được tạo ra lại được xử lý như thế hệ trước (xác định độ thích nghi và tạo thế hệ mới) cho đến khi có một cá thể đạt được giải pháp mong muốn hoặc đạt đến thời gian giới hạn.

Tuy nhiên đây không phải là thuật toán tìm kiếm đa điểm đơn thuần vì các điểm có tương tác với nhau theo nguyên lí tiến hóa tự nhiên. Trong ngữ cảnh sử dụng giải thuật di truyền, người ta có thể dùng khái niệm “cá thể” tương đương với khái niệm “giải pháp” các bước cơ bản của giải thuật được mô tả như sau:

* 1. Độ thích nghi tiêu chuẩn

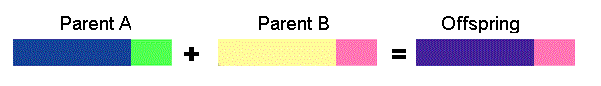
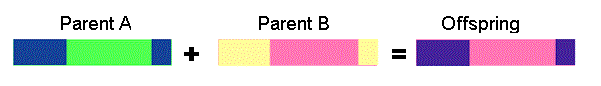
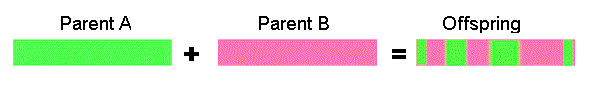
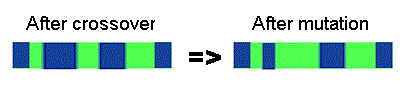
Hàm mục tiêu là hàm dùng để đánh giá độ tốt của cá thể. Hàm mục tiêu

nhận vào một tham số là gen của một cá thể và trả ra một số thực. Tùy theo giá

trị của số thực này mà ta biết độ tốt của cá thể đó.

* 1. Các phương pháp chọn lọc

Chọn lọc cá thể thông qua kết quả, hay mục đích của vấn đề dựa trên mức độ thích nghi của cá thể để tìm ra cá thể tốt nhất. Thông thường , đặt mỗi giá trị nhỏ tương ứng với một giá trị điểm thích nghi, kết quả đánh giá gồm tổng các điểm đó. Cá thể tốt nhất có điểm thấp nhất hoặc lớn nhất

* Chọn lọc Roulete: các cá thể được chọn theo độ thích nghi của chúng. Nhiễm sắc thể tốt hơn có cơ hội cao hơn để tham dự vào thế hệ tiếp theo.
* Chọn lọc xếp hạng: phương pháp này sẽ sắp hạng cá thể dựa trên độ thích nghi của chúng. Cá thể xấu nhất có giá trị 1, kết tiếp là 2… Và cá thể tốt nhất có độ thích nghi N (N là số cá nhiễm sắc thể có trong quần thể).
* Chọn lọc cạnh tranh: Chọn ra ngẫu nhiên 2 hoặc 3 nhiễm sắc thể khác nhau và so sánh với nhiễm sắc thể tồn tại. Nếu nhiễm sắc thể nào nhỏ hơn nhiễm sắc thể tồn tại thì sẽ bị loại ra khỏi quần thể. Quá trình này lặp lại đến hết N nhiễm sắc thể còn lại.
  1. Các phương pháp lai ghép
* Lai ghép một vị trí: từ hai nhễm sắc thể cha mẹ ban đầu ta cắt ở một vị trí sau đó ghép lại với nhau thành nhiễm sắc thể con.
* Lai ghép 2 vị trí: từ hai nhiễm sắc thể cha mẹ ba đầu ta cắt ở hai vị trí sau đó ghép chúng lại thành nhiễm sắc thể con.
* Lai ghép đồng dạng: những gen được copy ngẫu nhiễn nhiễm sắc thể thứ nhất sang nhiễm sắc thể thứ 2 và ngược lại.
  1. Các phương pháp đột biến
* Chọn ngẫu nhiễn một đoạn gen sau đó chèn vào nhiễm sắc thể cha, tạo ra nhiễm sắc thể mới.
* Để áp dụng thuật toán này vào bài toán xếp kịch cho một bệnh viện thì ta cần thực hiện những bước sau:

**Bước 1**: Khởi tạo dữ liệu bác sĩ, phòng khám, số ca.

**Bước 2:** Nhập, chỉnh sửa các ràng buộc.

**Bước 3:** Khởi tại quần thể.

**Bước 4:** Chọn 2 cá thể ngẫu nhiên cho giao đấu với nhau (nếu cá thể nào thắng cho vào quần thể mới).

**Bước 5:** Chọn ngẫu nhiên 2 cá thể trong quần thể mới cho lại ghép với nhau.

**Bước 6:** Chọn ngẫu nhiên một cá thể trong quần thể và cho đột biến.

**Bước 7:** Chọn ra quần thể tốt nhất.

**Bước 8:** Chỉnh sửa lại lịch bằng tay.

II. BÀI TOÁN SẮP XẾP LỊCH TRỰC BỆNH VIỆN ĐA KHOA

1. Giới thiệu bài toán

Bài toán sắp xếp lịch là một bài toán khó, sự phức tạp của bài toán không chỉ ở vấn đề tìm lịch trực cho bác sĩ thỏa mãm các ràng buộc về thời gian, ràng buộc chuyên khoa, phải thỏa mãn các điều kiện, yêu cầu về thời gian, không phòng trực nào được trống

Hiện nay, việc sắp lịch này ở hầu hết các bệnh viện đều được thực hiện một cách thủ công, và hầu như phải dựa vào kinh nghiệm thực tế mới có thể làm được. Thông thường việc sắp xếp thời lịch này phải mất một khoảng thời gian khá lâu.

Vậy bài toán đặt ra vấn đề cần sắp xếp lịch trực cho một bệnh viện. Cần có sự sắp xếp lịch trựuc cho các bác sĩ sao cho vừa phù hợp lại vừa tiện dụng nhất.

1. Dữ liệu bài toán

* Danh sách bác sĩ
* Danh sách phòng trực
* Danh sách ca trực
* Bảng yêu cầu ràng buộc

1. Các ràng buộc của bài toán

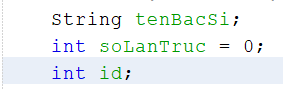
Ràng buộc cứng:

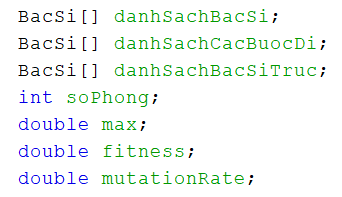
* + Một bác sĩ chỉ được trực 1 ca/ngày.
  + Một bác sĩ không trực 2 phòng/ca
  + Mỗi phòng luôn có bác sĩ trực

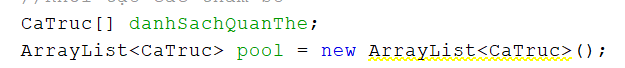
Ràng buộc mềm:

* + Tất cả các bác sĩ có cùng số lượng ca trực như nhau.
  + Không sắp xếp bác sĩ xin nghỉ phép vào ca trực.

1. Áp dụng thuật toán giải thuật di truyền vào bài toán
   1. Khởi tạo các class cho bài toán

* Chúng ta sẽ chia bài toán ra thành các class nhỏ, mỗi class thực hiện một nhiệm riêng.
*  Đầu tiên chúng ta khỏi tạo class BacSi gồm 3 tham số gồm mã bác sĩ, tên bác sĩ và một tham số dùng để đếm xem bác sĩ đó trực bao nhiêu lần trong tháng. Nhờ đó tất cả bác sĩ đều trực với số lượng bằng nhau.

- Class CaTruc gồm có danh sách các bác sĩ, danh sách bác sĩ trực trong 1 ngày, tên phòng, fitness, mutatiton.

* Quần thể gồm danh sách các cá thể.
  1. Khởi tạo quần thể

Trước khi tạo quần thể ban đầu trong phần này, chúng ta phải chuẩn bị sẵn về dữ liệu cho quá trình thực thi, từ lúc khởi tạo đến khi cho ra kết quả, bao gồm đầy đủ thông tin của một lớp đang được chọn. Tất cả như sau :

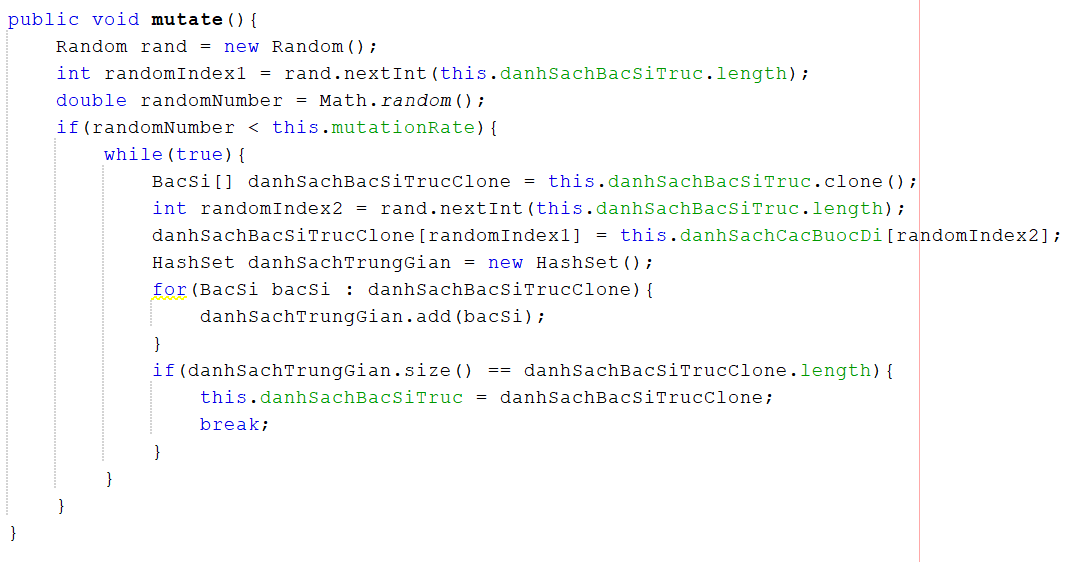
* Các ràng buộc .
* Các bác sĩ
* Số lượng cá thể
* Độ đột biến
* …

Hàng loạt các cá thể được tạo ra bằng cách random và được xem như quần thể ban đầu trong mô hình thuật giải di truyền của phần xếp lịch. Sau khi quần thể có đủ số lượng, bước tiếp theo là đánh giá quần thể, kiểm tra xem độ thích nghi tốt nhất hiện đang tồn tại của quần thể.

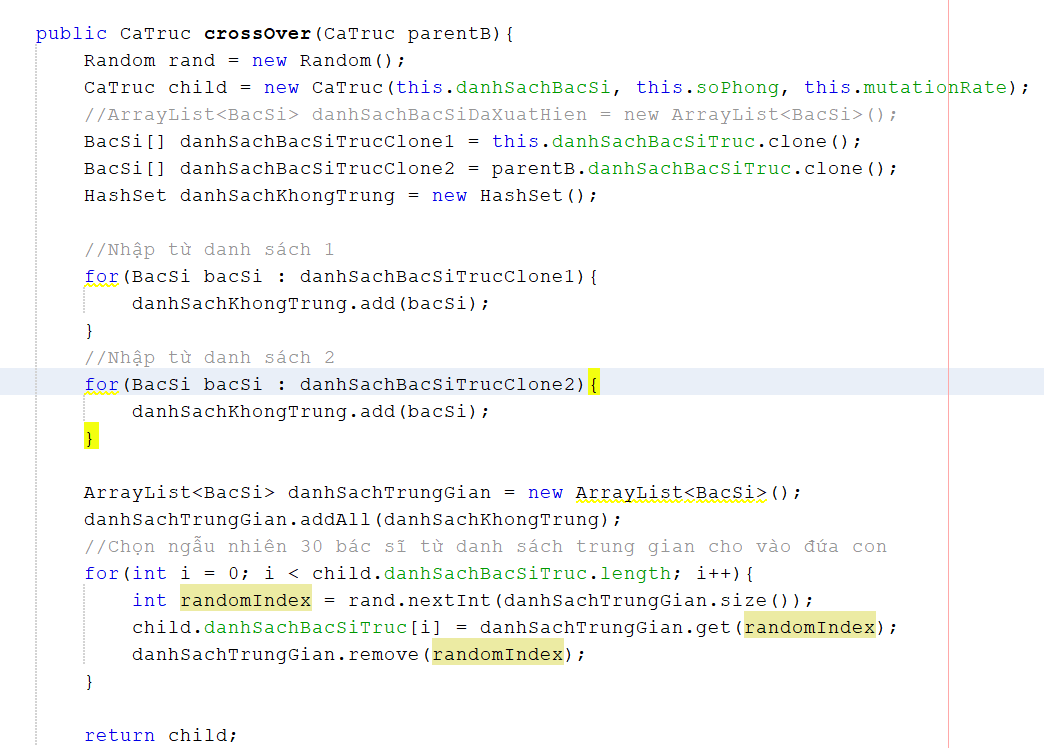
* 1. Lai ghép và đột biến
     1. Lai ghép

Về phần lai ghép thì ta random ra ngẫu nhiên ra 2 cá thể cho giao đấu với nhau. Cá thể nào mạnh hơn chúng ta sẽ cho vào một danh sách quần thể mới và để biết cá teher nào mạnh hơn thì ta có hàm calcFitness hàm này dùng để xét xem cá thể đó vi phạm các ràng buộc nhiều hay ít. Nếu vi phạm ít hơn sẽ được nhận vào quần thể mới.

Phần này áp dụng thực thi cho tất cả các lớp, tương ứng với mỗi lớp sẽ có một file lưu trữ tất cả các lịch lớp mà có thể sử dụng, dưới hình thức các nhiễm sắc thể trong quẩn thể. Ngoài mục đích xem xét kiểm tra, chúng còn được dùng làm thông tin để chạy lịch sau này. Sau khi ta cho giao đấu với nhau xong thì ta có một quần thể mới và để quần thể đó đa dạng hơn, độ thích nghi tốt hơn thì ta cho đột biến.

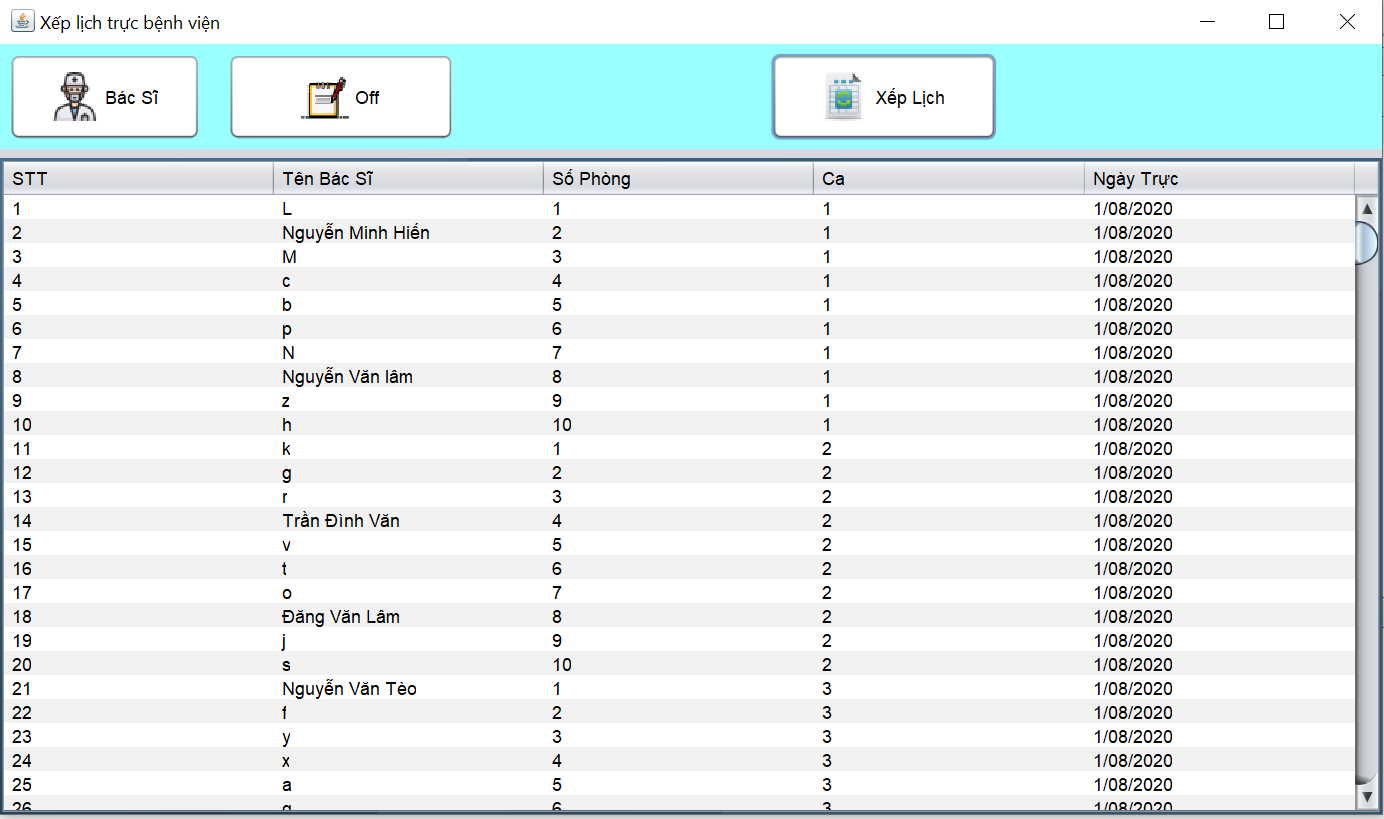


* + 1. Đột biến

ở phần đột biến này thì ta sẽ chọn ngẫu nhiên ra một cá thể ở trong quần thể. Và random ra một con số ngẫu nhiên. Nếu con số đấy nhỏ hơn tỷ lệ đột biến mà ta đưa vào thì ta sẽ đột biến ngẫu nhiên một ngày trong cá thể đó.

1. Chọn điểm dừng thuật toán

Một số bài toán chọn điểm dừng theo số thế hệ, hoặc dựa trên tính tương đối của kết quả, nhưng với bài toán này cần có một kết thúc tuyệt đối tốt nhất, mặt dù số thế hệ vẫn phải được chọn trước ngay từ đầu. Vì tính chất yêu cầu trong bài toán này là không bị sai lệch.

Nếu trong quá trình thực thi qua các giai đoạn, chỉ cần một kết quả không đạt đến điểm dừng, xem như bài toán sẽ không có kết quả.

III. KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

1. Kết luận.

* Đây vừa là bài toán lập nhiều kế hoạch có liên quan đến nhau, vừa là bài toán cấp phát tài nguyên. Để giải quết vấn đề này, chúng em đã mô hình hóa toán học bài toán. Từ đó dùng giải thuật di truyền với biểu diễn nhiễm sắc thể số nguyên để xây dựng phần mềm hỗ trợ ra quyết định.

1. Hạn chế

* Trong bài toán phần chọn 2 cá thể giao đấu và phần đột biến. Chưa được tối ưu vì ta sử dụng random.
* Phần ràng buộc chưa làm được đó là không xếp lịch cho những bác sĩ đăng kí nghỉ vào ngày đó.

1. Hướng phát triển.

Sẽ khác phục hạn chế trên và đột biến ở đoạn gen nào là phù hợp nhất chứ không phản random nữa. Sẽ phát triển ứng dụng cho cá nhân hóa cho người dùng hơn, không những xếp lịch trực mà còn xếp lịch học cho một trường học.

IV. TÀI LIỆU THAM KHẢO

[1] AI\_production\_of\_a\_school\_schedule (<HTTP://github.com/Aggeloss/AI_production_of_a_school_schedule>

[2] Genetic Algorithms (<https://www.geeksforgeeks.org/genetic-algorithms/>)

[3] Burke, E. K., Kingston, J. H., & de Werra, D. (2004d). Applications to timetabling. In J. Gross & J. Yellen (Eds.), The handbook of graph theory (pp. 445–474). London: Chapman Hall/CRC.

[4] E. K. Burke, D. G. Elliman, P. H. Ford, and R. F. Weare, “Examination timetabling in british universities a survey,” in The Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference (Lecture Notes in Computer Science 1153), E. Burke and P. Ross, Eds. Berlin, Germany: SpringerVerlag, 1996, pp. 76–90.

[5] Burke, E. K., & Petrovic, S. (2002). Recent research directions in automated timetabling. European Journal of Operational Research, 140(2), 266–280.

[6] Burke, E. K., Elliman, D. G., & Weare, R. F. (1994). A genetic algorithm for university timetabling. In Proceedings of the AISB workshop on evolutionary computing, University of Leeds, UK, 11–13 April 1994.

[7] Colijn, A. W., & Layfield, C. (1995b). Interactive improvement of examination schedules. In E. K. Burke & P. Ross (Eds.), Proceedings of the 1st international conference on the practice and theory of automated timetabling (pp. 112–121), 30 August–1 September 1995. Edinburgh: Napier University.

[8] Corne, D., Ross, P., & Fang, H. (1994). Evolutionary timetabling: Practice, prospects and work in progress. In P. Prosser (Ed.), Proceedings of UK planning and scheduling SIG workshop.

[9] A. Konar, Computational Intelligence: Principles, Techniques and Applications, Springer, 2005.

[10] Ross, P., Hart, E., & Corne, D. (1998). Some observations about GAbased exam timetabling. In E. K. Burke & M. W. Carter (Eds.), Lecture notes in computer science: Vol. 1408. Practice and theory of automated timetabling II: selected papers from the 2nd international conference (pp. 115–129). Berlin: Springer.