Tạp chí Tin học và Điều khiển học, T.17, S.2 (2001), 87–96

MỘT GIẢI PHÁP TIẾN HÓA CHO BÀI TOÁN THỜI KHÓA BIỂU

HOÀNG XUÂN HUẤN, NGUYỄN VIỆT THẮNG

Abstract. Timetable problem is a polular but NP hard-problem. Many practical applications have shown that genetic algorithms can be used effectively to solve this problem. Getting a convenient genotype and an initialization procedure for considered problem is the main obstacle to apply this method. In this paper, we introduce an evolutionary solution for the problem in schools which is easy to use.

Tóm tắt. Lập thời khóa biểu là bài toán phổ biến nhưng thuộc dạng NP khó, nan giải với những giải thuật thông thường. Tuy nhiên, sự phát triển và những ứng dụng thực tế của giải thuật di truyền đã cho thấy đây có thể là một phương pháp hiệu quả để giải quyết bài toán. Trở ngại chính trong việc áp dụng giải thuật vào bài toán chính là việc lựa chọn được một kiểu gene thích hợp và một thủ tục khởi tạo tương ứng. Trong bài này, chúng tôi sẽ giới thiệu một phương án tiến hóa cho bài toán lập thời khóa biểu tại các trường phổ thông, có thể dễ dàng đưa vào sử dụng.

1. MỞ ĐẦU

Lập thời khóa biểu là một bài toán, được nhiều người quan tâm trong vận trù học. Tuy vậy, nó thuộc loại bài toán NP khó (xem [4]) với nhiều loại ràng buộc phức tạp, nên khó giải quyết bằng các thuật truyền thống, đặc biệt khi có nhiều lớp và giờ học.

Trong thực hành đã chứng minh rằng giải thuật di truyền và các ứng dụng của nó với tên gọi chung là tính toán tiến hóa (xem [1,6]) là một phương pháp thực hành có hiệu quả để giải quyết các bài toán thực tiễn phức tạp. Trong thập kỷ qua, nhiều tác giả, mở đầu là các nhóm của Coloni [3] và Peacher [7] đã áp dụng có hiệu quả phương pháp này cho bài toán thời khóa biểu. Tuy vậy việc xây dựng một phần mềm theo các công trình đã công bố vẫn rất khó khăn, nên đến nay trong nước vẫn chưa có phần mềm nào được sử dụng rộng rãi trong các trường học.

Trong bài báo này chúng tôi giới thiệu một lược đồ tính toán tiến hóa dễ sử dụng để giải quyết bài toán. Trong đó thủ tục tạo mẫu đơn giản và giải quyết được nhiều ràng buộc khó, phần mềm thử nghiệm cho kết quả khả quan.

Trước hết, chúng tôi giới thiệu tóm tắt về giải thuật di truyền và tính toán tiến hóa, chi tiết hơn xem [1,6].

2. GIẢI THUẬT DI TRUYỀN VÀ TÍNH TOÁN TIẾN HÓA

Giải thuật di truyền (genetic algorithm viết tắt là GA) là các kỹ thuật phỏng theo quá trình tiến hóa tự thích nghi của các quần thể sinh học dựa trên học thuyết Darwin. Thoạt tiên, nó được sử dụng để giải quyết các bài toán riêng rẽ xuất phát từ sinh học vào cuối những năm 50 và được Holland trình bày một cách có hệ thống trong [5] để giải quyết bài toán tối ưu hàm nhiều biến nhờ kiểu gene nhị phân (nay được gọi là GA cổ điển). Nó nhanh chóng được nhiều tác giả cải tiến một cách phong phú để giải quyết các bài toán khó trong thực tiễn với tên gọi chung là tính toán tiến hóa (xem [1]). Việc giải quyết các bài toán tuy đa dạng nhưng thủ tục áp dụng vẫn dựa trên lược đồ GA cổ điển.

2.1. Giải thuật di truyền cổ điển

GA cổ điển được Holland giới thiệu (chi tiết hơn xem [5, 6]) để giải bài toán tối ưu:

$$\max\{f(x) \mid x \in M\}.$$

Ở đây M là hình hộp trong không gian số thực n-chiều, $f(x)$ dương với mọi x thuộc M.

Thủ tục GA được thực hiện như sau:

- Mỗi x trong M được mã hóa tương ứng bởi một xâu nhị phân độ dài m: $z = (z_1, \ldots, z_m)$ gọi là nhiễm sắc thể (còn gọi là cá thể), mỗi z_i được gọi là một gene. Xây dựng thủ tục mã hóa, giải mã tương ứng.
- Xác định hàm *eval* trên tập trên tập nhiễm sắc thể để đánh giá độ "thích nghi" của mỗi cá thể: $eval(z) = f(x)$, trong đó x là vectơ tương ứng với z.
- Tạo quần thể ban đầu $P(0)$ gồm N phần tử và thực hiện quá trình tiến hóa theo cấu trúc:

```
Procedure GA
Begin
    t ← 0;
    Khởi tạo P(t);
    Đánh giá P(t);
    Repeat
        t ← t + 1;
        Chọn lọc Q(t) từ P(t − 1);      // nhờ bánh xe xổ số,
        Tái tạo P(t) từ Q(t);           // nhờ các toán tử di truyền,
        Đánh giá P(t) và chọn cá thể tốt nhất;
    Until điều_kiện_kết_thúc,
    Biểu diễn lời giải;
End;
```

Các thủ tục chọn lọc môt quần thể theo phương pháp *bánh xe xổ số* và tái tạo nhờ các *toán tử di truyền* được thực hiện như sau.

2.1.1. Thủ tục chọn lọc

Với mỗi quần thể $P(t-1)$ gồm N nhiễm sắc thể: $P(t-1) = \{v_1, \ldots, v_N\}$ ta xây dựng *bánh xe xổ số* và thực hiện quá trình chọn lọc:

- Bánh xe xổ số

  Đánh giá độ phù hợp toàn phần: $F = \sum_{i=1}^{N} \text{eval}(v_i)$.

  Tính các sác xuất chọn p_i của nhiễm sắc thể v_i: $p_i = \text{eval}(v_i)/F$.

  Tính các xác suất tích lũy q_i của các nhiễm sắc thể v_i: $q_i = \sum_{j=1}^{i} p_j$.

- Quá trình chọn lọc

  Quá trình chọn lọc quần thể $Q(t)$ từ $P(t-1)$ dựa vào bánh xe xổ số được thực hiện theo cách sau:

  Đối với mỗi số tự nhiên $k \in \{1, \ldots, N\}$ tạo một số ngẫu nhiên $r_k \in [0, 1]$.

Nếu $q_i \geq r_k > q_{i-1}$ thì chọn v_i thuộc $Q(t)$. Hiển nhiên, ở đây mỗi nhiễm sắc thể có thể được chọn nhiều lần và $Q(t)$ vẫn có N phần tử. Các cá thể v có độ thích nghi $eval(v)$ lớn sẽ có khả năng được chọn nhiều hơn.

2.1.2. Quá trình tái tạo

Quá trình tái tạo dựa trên các toán tử di truyền: *tương giao chéo và biến dị*.

- **Các toán tử di truyền**

  Toán tử tương giao chéo: Với 2 nhiễm sắc thể

$$x = x_1, \ldots, x_m) \text{ và } y = (y_1, \ldots, y_m).$$

Chọn điểm tương giao k (có thể ngẫu nhiên) ta sẽ sinh được hai nhiễm sắc thể mới:

$$x' = x_1, \dots, x_k, y_{k+1}, \dots, y_m)) \text{ và } y' = (y_1, \dots, y_k, x_{k+1}, \dots, x_m).$$

Toán tử biến dị: Nếu gene x_k của nhiễm sắc thể $x = (x_1, \dots, x_m)$ biến dị thì ta được nhiễm sắc thể mới x' có:

$$x' = x_1, \dots, x_{k-1}, 1-x_k, x_{k+1}, \dots, x_m).$$

- **Thủ tục tái tạo**

Cho trước các xác suất tương giao chéo p_c và xác suất biến dị p_m.
Đối với mỗi nhiễm sắc thể v_i (i chạy từ 1 đến N) thuộc $Q(t)$, ta tạo ra một số ngẫu nhiên $r \in [0, 1]$. Nếu $r < p_c$ thì v_i được đưa vào tập tương giao chéo. Tập này được chia thành cặp, nếu lẻ thì có thể thêm hoặc bớt ngẫu nhiên một nhiễm sắc thể khác và áp dụng toán tử tương giao chéo để tạo nên hậu duệ mới thay thế cho nó.

Sau khi tương giao chéo, đối với mỗi gene của mỗi nhiễm sắc thể ta tạo một số ngẫu nhiên $r \in [0, 1]$. Nếu $r < p_m$ thì gene này được biến dị.

Quá trình trên cho ta quần thể $P(t)$ của thế hệ t và được đánh giá để chọn phần tử có độ thích nghi tốt nhất.

2.1.3. Cơ sở toán học của GA

Các đánh giá về sự hội tụ của GA còn rất nghèo (xem [1, 2, 6]). Các kết quả đạt được chủ yếu dựa trên định lý về lược đồ chứng minh sự hội tụ theo xác suất tới lời giải tối ưu của bài toán. Tuy nhiên, về mặt thực hành, giải thuật di truyền vẫn là một giải thuật được ưa thích để giải các bài toán khó trong thực tế và cho lời giải đủ tốt. Đối với các bài toán đã có phương pháp giải tốt bằng pháp truyền thống thì GA kém hiệu quả hơn.

2.2. Tính toán tiến hóa

Khi bài toán có miền chấp nhận được lớn trong không gian nhiều chiều thì độ rộng của mỗi nhiễm sắc thể lớn nên việc áp dụng GA cổ điển rất khó khăn, đặc biệt khi có các ràng buộc phức tạp thì các toán tử di truyền theo kiểu đã nêu tỏ ra kém hiệu quả sử dụng. Hàng loạt các phát triển phong phú của GA cổ điển về kiểu gene, cấu trúc nhiễm sắc thể và các toán tử di truyền đã được đề xuất và ứng dụng có hiệu quả để giải các bài toán khác nhau trong vận trù học. Mặt khác để tìm lời giải cho các bài toán khó trong thực tiễn, người ta đưa ra các hàm "đích" đo độ "thích nghi" của mỗi lời giải tiềm năng và áp dụng GA để tìm các lời giải. Các phát triển đó có các tên gọi khác nhau chẳng hạn:

Chiến lược tối ưu: Nhằm giải quyết các bài toán tối ưu rời rạc hoặc liên tục khó và tối ưu tham số. Trong đó các kiểu gene khác nhau được sử dụng để xử lý ràng buộc và giảm khối lượng xử lý dữ liệu.

Lập trình tiến hóa: Ứng dụng GA trong trí tuệ nhân tạo nhờ sử dụng các ôtômat hữu hạn để tạo ra các chương trình thích ứng với yêu cầu, giúp tạo hành vi cho các robot hoặc agent thông minh.

Chương trình tiến hóa: Ứng sụng GA để tìm lời giải cho các bài toán khác nhau khi không gian tìm kiếm phức tạp. Chúng có tên gọi chung là phương pháp tính toán tiến hóa (evolutionary computation - viết tắt là EC). Lược đồ chung của thuật toán tiến hóa là:

- Chọn một kiểu gene và cấu trúc nhiễm sắc thể thích hợp cho các lời giải tiềm năng của bài toán. Xây dựng chủ tục chuyển đổi giữa chúng.
- Đưa ra một hàm để đo "độ tốt" của các lời giải tiềm năng nhờ đó xác định hàm đích cho EC.
- Xác định các toán tử di truyền (tương giao chéo và biến dị) thích hòp cho từng bài toán và các ràng buộc của chúng. Các toán tử có thể nhiều để vận dụng thích hợp xử lý ràng buộc.
- Xây dựng thủ tục tạo quần thể ban đầu và lặp nhiều lần quá trình chọn lọc, tái tạo để nhận được lời giải.

Có thể mô tả thuật toán như sau:

```
Proceduce EC;
Begin
    t ← 0;
    Khởi tạo P(t)                      // khởi tạo quần thể
    Đánh giá P(t);                     // đánh giá độ thích nghi
    while not kết_thúc do // vòng lặp tiến hóa
    begin
        P'(t) ← Biến đổi (P(t));           // biến đổi quần thể
        Đánh giá P'(t);                    // đánh giá độ thích nghi mới
        P(t + 1) ← Chọn lọc (P'(t));       // tạo ra thế hệ con mới
        t ← t + 1;
    end;
End;
```

Để sử dụng thuật toán tiến hóa, khó khăn chính là chọn lọc được kiểu gene, cấu trúc nhiễm sắc thể và các thủ tục tạo mẫu, toán tử tử di truyền thích hợp để xử lý các ràng buộc. Còn các khó khăn trong xây dựng phần mềm thể hiện giải thuật là tổ chức dữ liệu.

3. MỘT LƯỢC ĐỒ TIẾN HÓA CHO BÀI TOÁN THỜI KHÓA BIỂU

Có nhiều loại bài toán thời khóa biểu, chúng tôi xét bài toán trong trường học.

3.1. Phát biểu bài toán

Bài toán tổng quát được phát biểu như sau:

Một danh sách xác định các lớp học, mỗi lớp có một danh sách xác định các giờ học trong một tuần bao gồm môn học, tên giáo viên và số tiết. Các lớp học được phân bố trong các phòng học đã biết.

Tìm một phương án phân bố giờ học, môn học cho các lớp thỏa mãn một số ràng buộc ngặt (bắt buộc) và các ràng buộc mềm (theo sở thích cá nhân, nếu được thì càng tốt, nhưng không bắt buộc).

Có thể nêu ra một số ràng buộc phổ biến cần giải quyết trong trường phổ thông:

Ràng buộc ngặt

- Một giáo viên trong một tiết dạy không dạy quá một lớp.
- Một lớp trong một tiết học có không quá một giáo viên.
- Một lớp trong một tiết học không quá một môn.
- Không được xếp lịch dạy vào các giờ bận của giáo viên.
- Một số môn không được dạy quá k tiết trong một ngày.
- Trong mỗi buổi học ở mỗi lớp các tiết học liên tục.
- Trong một buổi học, các tiết của cùng một môn không được tách rời.
- Giáo viên chỉ phải dạy một buổi (hoặc lượng giờ hạn chế) trong một ngày.
- Một số môn phải phân vào các giờ xác định (ví dụ giờ cuối của ngày cuối tuần phải là giờ sinh hoạt lớp).

Các ràng buộc mềm

- Có các giáo viên thích dạy hoặc nghỉ vào các giờ nhất định.
- Các giờ dạy của giáo viên trong một buổi phân bố liên tục.
- Các tiết học của một môn trong tuần phân bố càng đều càng tốt.
- ....

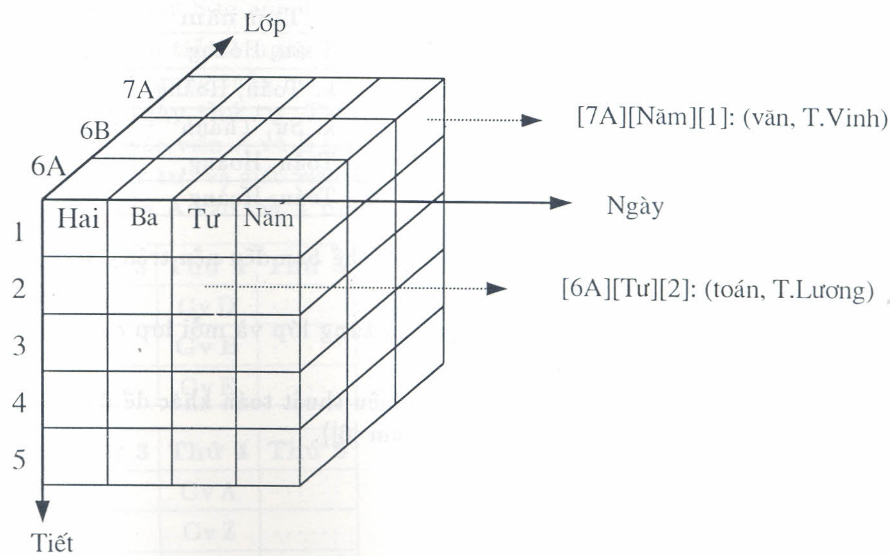
3.2. Giải thuật tính toán tiến hóa

Trong mục này chúng tôi giới thiệu một lược đồ tính toán tiến hóa cho bài toán thời khóa biểu, một thể hiện của thuật toán được giới thiệu ở mục 4.

3.2.1. Cấu trúc nhiễm sắc thể và kiểu gene

Nhiễm sắc thể có cấu trúc mảng 3 chiều (xem hình 3) mỗi vị trí gene trên nhiễm sắc thể được xác định bởi 3 tham số [**lớp, ngày, tiết**]. Mỗi gene trên nhiễm sắc thể mã hóa cho một tiết học trong một ngày của một lớp trong tuần và có dạng bản ghi (**môn, giáo viên**) ghi mã số **tên giáo viên** giảng dạy và **môn học** tương ứng với **tiết** vào **ngày tương ứng của lớp**.

Khi nhìn theo một nhát cắt của chiều "**lớp**" ta sẽ có một thời khóa biểu thông thường cho lớp tương ứng và mỗi nhiễm sắc thể được xem là các thời khóa biểu của các lớp được xếp chồng lên nhau.



Hình 1. Cấu trúc một nhiễm sắc thể

3.2.2. Khởi tạo quần thể ban đầu

Để khởi tạo quần thể ban đầu $P(0)$ trước hết ta phân bố số tiết trong mỗi ngày cho mỗi lớp. Chúng có thể xác định trước bởi người sử dụng, khi người dụng không yêu cầu thì được phân bố một cách tự động theo các nguyên tắc định sẵn và sau đó cố định trong quá trình tiến hóa. Chẳng hạn: lớp 7A có 23 tiết một tuần, học từ thứ hai tới thứ sáu khi phân tự động sẽ có 2 ngày 4 tiết và 3 ngày 5 tiết. Các tiết học phân bố liên tục từ giờ đầu (hoặc định trước) trong ngày học của lớp.

Bây giờ thủ tục tạo ngẫu nhiên một nhiễm sắc thể được thực hiện như sau.

Mỗi nhiễm sắc thể được xét như là hợp các thời khóa biểu tương ứng của mỗi lớp (xét theo nhát cắt **lớp** đã nói ở trên).

– Xét một lớp cho trước, giả sử mỗi tuần có n giờ học đã được phân bố. Ta đánh số các giờ học từ 1 đến n. Các tiết theo môn học của lớp cũng được đánh số từ 1 đến n. Khi có các tiết học yêu cầu phân cố định thì phân trước và loại khỏi danh sách.

– Tạo một vectơ ngẫu nhiên (n-1)-chiều $(m_1, \dots, m_{n\text{-}1})$, trong đó m_k là số tự nhiên thuộc tập $\{1, \dots, n-k+1\}$ với mọi k từ 1 đến $(n-1)$. Khi đó giả sử $k-1$ giờ học có thứ tự đầu đã được phân tiết thì loại chúng ra khỏi danh sách tương ứng và còn lại một danh sách được đánh số từ 1 đến $(n-k+1)$ và ta phân giờ học thứ tự m_k trong danh sách này vào giờ thứ k.

Ví dụ: Xét lớp 7A với giờ học

Thứ hai	Thứ ba	Thứ tư	Thứ năm	Thứ sáu
1. Toán, Hoàng	6. Văn, Lê	10. Địa, Toàn	13. Toán, Hoàng	16. Thể dục, Lan
2. Toán, Hoàng	7. Văn, Lê	11. Sử, Thanh	x. Toán, Hoàng	17. GDCD, An
3. Toán Hoàng	8. Văn, Lê	12. Văn, Lê	x. Sử, Thanh	18. Anh, Hòa
4. Toán Hoàng	9. Văn, Lê	x. Văn, Lê	14. Sinh, An	19. Anh, Hòa
5. SH, Thu	x	x. Lý, Thanh A	15. Lý, Thanh A	x

Số thứ tự môn học, giờ học đánh số như trên, các ô x là đã phân trước, không đưa vào danh sách.

Khi đó vectơ ngẫu nhiên: (17, 15, 4, 8, 14, 6, 12, 8, 9, 8, 7, 5, 3, 2, 1, 3, 2, 1) ta có:

Thứ hai	Thứ ba	Thứ tư	Thứ năm	Thứ sáu
GDCD, An	Văn Lê	Văn, Lê	Toán, Hoàng	Sinh, An
Lý, Thanh A	Thể dục, Lan	Địa, Toàn	x. Toán, Hoàng,	Văn, Lê
Toán, Hoàng	Sử, Thanh	Văn, Lê	x. Sử, Thanh	Sinh hoạt, Thu
Văn, Lê	Toán, Hoàng	x. Văn, Lê	Toán, Hoàng,	Anh, Hòa
Anh, Hòa		x. Lý, Thanh A	Toán, Hoàng	

Cấu trúc nhiễm sắc thể và thủ tục khởi tạo quần thể ban đầu nêu trên đáp ứng các ràng buộc ngặt:

- Đủ số tiết trong một tuần của mỗi giáo viên cho từng lớp và mỗi lớp có đủ số môn và số tiết trong một tuần.
- Mỗi giờ chỉ có một môn học ở mỗi lớp (trong nhiều thuật toán khác để đảm bảo được yêu cầu này phải dùng tới một thuật toán phân phối (xem [6]).
- Thỏa mãn ngay các yêu cầu về các giờ cố định.

3.2.3. Xác định đích (đo độ thích nghi)

Hàm đích f cần nhận giá trị dương và có thể thay đổi theo từng thế hệ (bước lặp) để tăng hiệu quả thuật toán, đáp ứng được các ràng buộc đa dạng. Nếu bài toán có k loại ràng buộc thì giá trị của nó có thể xác định đối với mỗi cá thể v như sau:

$$f(v) = M + \sum_{i=1}^{k} g_i(v).$$

Trong đó M là số cho trước, $g_i(v)$ là các hàm đánh giá theo ràng buộc i của v. Chẳng hạn:

$g_1(v) = -Ax$ (đánh giá số tiết học trùng của giáo viên, x là số tiết học bị trùng giờ.

$g_2(v) = -By$ (đánh giá số tiết học trùng vào giờ bận của giáo viên, y là số tiết dạy bị bận); A, B là tham số cho trước....

3.2.3. Các toán tử di truyền

Ta có thể dùng nhiều loại toán tử biến dị để xử lý các loại ràng buộc (phỏng theo việc biến đổi gene có định hướng trong công nghệ sinh học) còn toán tử tương giao chéo thì tương tự như GA cổ điển.

Các toán tử biến dị

Toán tử đổi chỗ tiết học trong một lớp: đổi chỗ hai gene ở hai vị trí bất kỳ trong thời khóa biểu của một lớp (nhằm đổi chỗ tiết học của môn học trong ngày có nhiều tiết sang ngày khác không có hoặc ít tiết hơn), được minh họa trên hình 2.

	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	Toán	......	......	......
T2	......	......	Lý	......
T3	......	......	......	......

	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	Lý	......	......	......
T2	......	......	Toán	......
T3	......	......	......	......

Hình 2. Toán tử đổi chỗ tiết học trong một lớp

Toán tử đổi chỗ giáo viên (để giảm bớt các tiết trùng của giáo viên): Khi có một giáo viên A bị trùng giờ dạy trên hai lớp (giả sử là lớp S và lớp T) vào tiết học T1 của ngày N1, ta sẽ tìm một tiết học T2 vào ngày N2 trong tuần sao cho giáo viên đó không có giờ dạy (luôn tìm được). Đối chiếu lên lớp học đang dạy (ví dụ lớp S) ta tìm được giáo viên B dạy tiết học T2 ngày N2. Đổi chỗ giờ dạy của hai giáo viên tại lớp S ta xóa được xung đột tại tiết T1 của giáo viên (xem hình 3).

Ví dụ: Giáo viên A tại **tiết 1 ngày thứ hai** vừa có **tiết văn tại lớp 6A** vừa có **tiết Văn tại lớp 7A**. Ta tìm một tiết trong tuần mà giáo viên đó không phải dạy một lớp nào (luôn tìm được), giả sử tìm được **tiết 2 ngày thứ tư**. Tại **lớp 6A** vào **tiết 2 ngày thứ tư**, giáo viên B dạy Toán. Ta đổi giờ dạy của hai giáo viên A và B tại lớp 6A. Sau khi đổi ta có: giáo viên **A** dạy **Văn** tại **lớp 6A** vào **tiết 2 ngày thứ tư** và giáo viên **B** dạy **Toán** tại **lớp 6A** vào **tiết 1 ngày thứ hai**. Như vậy, sự trùng tiết dạy của **A** vào tiết **1 ngày thứ hai** đã bị loại bỏ.

L. 6A	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	**Gv A**	......	Gv D	......
T2	Gv C	......	**Gv B**	......
T3	Gv C	......	Gv E	......

L. 7A	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	**Gv A**	......	Gv A	......
T2	Gv X	......	Gv Z	......
T3	Gv Y	......	Gv Z	......

L. 6A	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	**Gv B**	......	Gv D	......
T2	Gv C	......	**Gv A**	......
T3	Gv C	......	Gv E	......

(**Thứ 2, tiết 1**) giáo viên **Gv A** bị trùng giờ.

(**Thứ 4, tiết 2**) giáo viên **Gv A** không có giờ, giáo viên **Gv B** dạy tại lớp 6A vào tiết đó. Đổi chỗ hai giáo viên.

Hình 3. Toán tử đổi chỗ giáo viên

Toán tử chuyển dịch tiết học trong một buổi: Khi các giờ giảng của một giáo viên bị tách rời ở một lớp trong một buổi ta xếp lại các tiết trong buổi đó của lớp, các tiết cách được dồn xuống cuối, các tiết của cùng một giáo viên được nối lại thành tiết học liên tiếp (xem hình 4).

	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	Toán	Văn	Tin	
T2	Hóa	Sử	Lý	
T3	Toán	Văn	Tin	Lý

	Thứ 2	Thứ 3	Thứ 4	Thứ 5
T1	Toán	Văn	Tin	Lý
T2	Toán	Văn	Tin	
T3	Hóa	Sử	Lý	

Hình 4. Toán tử chuyển dịch tiết học trong một ngày

Toán tử thay đổi thời khóa biểu toàn bộ lớp: khởi tạo lại thời khóa biểu ở một lớp ngẫu nhiên trong nhiễm sắc thể được xét theo thủ tục đã nêu ở 3.2.2.

Các toán tử tương giao chéo

Ta sẽ dùng hai toán tử tương giao chéo: toàn bộ và một phần.

Toán tử tương giao chéo toàn bộ: thực hiện đổi chỗ toàn bộ thời khóa biểu ở một lớp được chọn (có thể ngẫu nhiên) của hai nhiễm sắc thể tương giao.

Toán tử tương giao chéo một phần: thực hiện tương tự như toán tử trên nhưng cố định một vài môn học, và chỉ đổi chỗ những môn học còn lại.

3.2.4. Thủ tục tiến hóa

Quá trình tiến hóa được giữ nguyên như trong mục 2.2. Cụ thể là:

- Đầu tiên khởi tạo quần thể $P(0)$ với N phần tử theo thủ tục 3.2.2 và đánh giá độ thích nghi.
- Tại vòng lặp đời thứ t, quần thể $P(t)$ được tái tạo thành một quần thể trung gian $P'(t)$ nhờ vận dụng linh hoạt các toán tử di truyền (số lượng các cá thể trong $P'(t)$ có thể lớn hơn N).
- Thủ tục chọn lọc thực hiện theo phương pháp bánh xe xổ số như đã nêu (xem 4.5) để chọn $P(t+1)$ từ $P'(t)$.
- Cuối cùng $P(t+1)$ được đánh giá lại với các cá thể mới để kết thúc một vòng lặp.

4. TRƯỜNG HỢP THỬ NGHIỆM

Thuật toán trên đã được xây dựng phần mềm thử nghiệm lập thời khóa biểu cho trường trung học và được thử nghiệm bởi bộ dữ liệu thực tế lấy từ trường THCS Chu Văn An. Đây là bài toán khó nhất trong lớp bài toán lập thời khóa biểu với các lý do:

Thứ nhất: Các trường trung học cơ sở thường có nhiều lớp (30 - 50 lớp) với số lượng giáo viên lớn (60 - 90 người).

Thứ hai: Số tiết học của mỗi lớp trong một tuần khá dày (25 - 27 tiết) dẫn đến số tiết dạy của giáo viên trong tuần cũng rất lớn (nhiều khi hơn 20 tiết), các tiết của một môn học không được quá 2 tiết trong một ngày.

Thứ ba: Các trường trung cơ sở thường có những yêu cầu đặc biệt như tiết đầu tuần và cuối tuần là của giáo viên chủ nhiệm, giáo viên dạy sáng thì không phải dạy chiều.

Tất cả những điều này khiến việc xếp lịch dễ bị trùng lớp, trùng tiết và không gian tìm kiếm quá rộng. Một ví dụ đơn giản là một tuần học 25 tiết, có 40 lớp và mỗi lớp có 8 môn học thì không gian tìm kiếm là 8^{1000} trường hợp. Với không gian tìm kiếm như vậy không thể duyệt hết toàn bộ không gian tìm kiếm theo các giải thuật truyền thống.

4.1. Bài toán thời khóa biểu cho trường trung học cơ sở

Bài toán được xét như sau. Lập thời khóa biểu hàng tuần cho trường có M lớp học và N giáo viên; mỗi tuần học p ngày, mỗi ngày có hai buổi; mỗi buổi học tối đa q tiết. Mỗi giáo viên được phân công dạy các môn cụ thể với số tiết ở những lớp đã biết. Một lớp sẽ học ở các phòng học cố định vào những buổi đã biết.

Các ràng buộc được xét là:

Ràng buộc ngặt:

- Một giáo viên không dạy quá 1 lớp trong một tiết học, mỗi lớp học không quá 1 giáo viên trong một tiết.
- Mỗi giáo viên có một lịch các giờ bận. Không được xếp lịch giảng vào giờ bị bận đó.
- Không có tiết trống giữa các tiết học trong một lớp.
- Giờ giảng của giáo viên một môn ở một lớp trong mỗi buổi không bị tách rời.
- Giáo viên chỉ dạy một buổi trong mỗi ngày, một môn không vượt quá trong một buổi ở một lớp.
- Một số tiết được định trước bởi người lập lịch.

Ràng buộc mềm:

- Một số giáo viên có một số tiết định trước nếu không xếp lịch thì tốt.
- Giáo viên dạy liên tục trong mỗi buổi.

4.2. Thủ tục thực hiện

Cho trước các tham số $Pcrs1$, $Pcrs2$, $Pmut$ ...

```
Procedure EA_for_schedule;
Begin
    t ← 0
    Khởi tạo P(t);
    Đánh giá P(t);
    Repeat
        Số lần ← Random()
        For i ← 1 to Số_lần do
        begin
        Hệ_số ← Random()
            If Hệ_số < Pcrs1 then      Tương_giao_chéo_toàn_bộ(P'(t));
            Hệ_số ← Random()
            If Hệ_số < Pcrs2 then      Tương_giao_chéo_một_phần(P'(t));
            Hệ_số ← Random();
            If Hệ_số < Pmut1 then      Đổi_tiết_học(P'(t));
            Hệ_số ← Random()
            If Hệ_số < Pheu1 then      Đổi_giáo_viên(P'(t));
            Hệ_số ← Random()
            If Hệ_số < Pheu2 then      Chuyển_dịch_tiết(P'(t));
            Hệ_số ← Random();
            If Hệ_số < Pbdm then       Biến_dị_mạnh(P'(t));
        end;
        P(t + 1) ← Chọn_lọcP'(t);
        t ← t1;
    Until điều_kiện_kết_thúc;
End;
```

Trong đó $P'(t)$ bao gồm cả $P(t)$ và các phần tử mới được tái tạo, các cá thể được thực hiện tương giao chéo và biến dị có tính ngẫu nhiên tương tự thủ tục cổ điển.

4.3. Kết quả thử nghiệm

Chương trình được thử nghiệm với bộ dữ liệu của trường THCS Chu Văn An:

- Gồm 4 khối 6, 7, 8, 9 có tổng cộng 42 lớp được chia làm 2 buổi học (20 lớp sáng và 22 lớp chiều).
- Số môn học trung bình trong mỗi lớp một học kỳ là 12 môn. Số tiết học trung bình trong một tuần của mỗi lớp là 27 tiết học.
- Một tuần có 6 ngày học, mỗi ngày có tối đa 5 tiết học.
- Có tổng số 90 giáo viên. Giáo viên có số giờ nhiều nhất là 23 tiết một tuần.

Kiểm tra trên máy CELERON 333 MHz, 32 Mb RAM, chương trình chạy 100 vòng đời hết 02'30" và sau 20-30 phút thì không còn vi phạm ràng buộc ngặt. Số liệu thống kê được như sau:

Số vòng đời (vòng)	**Kết quả tốt nhất**			**Kết quả trung bình** (10 lần chạy)		
	Trùng GV	Trùng tiết	Giờ cách	Trùng GV	Trùng tiết	Giờ cách
100	0	0	1	0	2	4
150	0	0	1	0	1	1
200	0	0	0	0	0	1

5. KẾT LUẬN

Trên đây chúng tôi đã trình bày một phương pháp thực hiện tính toán tiến hóa cho bài toán thời khóa biểu. Thủ tục tạo mẫu và các toán tử di truyền dễ thực hiện, có thể ứng dụng rộng rãi để tạo nên các phần mềm thích ứng cho các loại thời khóa biểu khác nhau và có thể mở rộng ra cho một số bài toán xếp lịch khác. Ngoài ra, trong quá trình xây dựng phần mềm, việc phân bố các toán tử một cách thích hợp để vừa đảm bảo độ hội tụ vừa đảm bảo tính đa dạng cũng là một vấn đề cần quan tâm nghiên cứu giải quyết. Chúng tôi hy vọng trong thời gian tới sẽ có các sản phẩm phần mềm sử dụng rộng rãi nhờ ứng dụng phương pháp này.

TÀI LIỆU THAM KHẢO

[1] J. Back, U. Hammel, and H. P. Shwefel, Evolutionary computation: Comments on the history and current state, *IEEE Transactions on Evolutionary Computation* **1** (1) (1997) 3–17.

[2] R. Ceft, A new genetic algorithm, *Analysis of Applied Probability* **6** (3) (1996) 778–817.

[3] A. Colorni, M. Dorigo, and V. Manieggo, *Genetic Algorithm and Highly Constrained Problems, the Timetable Case, Problem Solving from Nature*, Springer-Verlag, Lecture Notes in Computer Science, Vol. 496, 1991, p. 55–59.

[4] S. Even, A. Itai, and A. Shamir, On the complexity of timetable and multicommodity flow problems, *SIAM Journal on Computing* **5** (4) (1976) 691–703.

[5] J. A. Holland, *Adaption in Natural and Artificial System*, University of Michigan press, Ann Arbor, 1975.

[6] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Berlin, Germany, Springer, 1996.

[7] B. Peacher, A. Luchian, and M. Petrius, Two solutions to the general timetable problem using evolutionary methods, *Proceeding of the Evolutionary Computational Conference*, Orlando, 26-29 June, 1994.

Nhận bài ngày 12 tháng 8 năm 2000

Khoa Công nghệ, Đại học Quốc gia Hà Nội.