TẠP CHÍ KHOA HỌC
TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HỒ CHÍ MINH
Tập 18, Số 2 (2021): 331-341
ISSN: 1859-3100
HO CHI MINH CITY UNIVERSITY OF EDUCATION
JOURNAL OF SCIENCE
Vol. 18, No. 2 (2021): 331-341
Website: *http://journal.hcmue.edu.vn*

Bài báo nghiên cứu

KHẢO SÁT CÁC THUẬT TOÁN ĐỊNH HƯỚNG KHOA HỌC MÁY TÍNH MÔN TIN HỌC CHƯƠNG TRÌNH GIÁO DỤC PHỔ THÔNG MỚI

Phan Tấn Quốc*, Phan Thị Kim Loan, Trương Tấn Khoa
Trường Đại học Sài Gòn, Việt Nam
Tác giả liên hệ: Phan Tấn Quốc – Email: quocpt@sgu.edu.vn
Ngày nhận bài: 20-3-2020; ngày nhận bài sửa: 29-4-2020; ngày duyệt đăng: 25-02-2021

TÓM TẮT

Chương trình giáo dục phổ thông mới sẽ được thực hiện từ năm học 2022-2023 đối với bậc trung học phổ thông; trong đó chương trình môn Tin học được thiết kế gồm nội dung cốt lõi và các chuyên đề học tập định hướng nghề nghiệp; hiện tại, yêu cầu cần đạt được của các nội dung cốt lõi và các chuyên đề học tập này được liệt kê ngắn gọn. Các chuyên đề học tập được thiết kế theo hai định hướng là Tin học ứng dụng và Khoa học máy tính; trong đó, định hướng Tin học ứng dụng tập trung vào việc sử dụng các phần mềm thông dụng thiết yếu để nâng cao hiệu suất công việc, tạo cơ hội cho học sinh làm ra sản phẩm số thiết thực phục vụ học tập và cuộc sống; định hướng Khoa học máy tính tập trung phát triển tư duy máy tính, năng lực phân tích bài toán, lựa chọn kiểu dữ liệu và thiết kế thuật toán. Trong bài báo này, chúng tôi khảo sát và chi tiết hóa nội dung các chuyên đề học tập định hướng Khoa học máy tính thuộc Chương trình Tin học lớp 11 bao gồm thuật toán đệ quy, thuật toán chia để trị và thuật toán quay lui; bài báo này nhằm đưa ra một góc nhìn về các thuật toán môn Tin học lớp 11 để các giáo viên trung học phổ thông tham khảo.

Từ khóa: thuật toán quay lui; thuật toán nhánh cận; thuật toán chia để trị; thuật toán quy hoạch động; thuật toán sinh; thuật toán tham lam; thuật toán đệ quy

1. Giới thiệu

1.1. Chương trình môn Tin học thuộc Chương trình giáo dục phổ thông hiện tại

Chương trình môn Tin học 10 hiện tại có các chủ đề: Một số khái niệm cơ bản của tin học; hệ điều hành; soạn thảo văn bản; mạng máy tính và Internet (Ho et al., 2014).

Chương trình môn Tin học 11 hiện tại có chủ đề: Khái niệm lập trình và ngôn ngữ lập trình; chương trình đơn giản; cấu trúc rẽ nhánh và lặp; kiểu dữ liệu có cấu trúc; tệp và và thao tác với tệp; chương trình con và lập trình có cấu trúc (Ho et al., 2014).

Chương trình môn Tin học 12 hiện tại có các chủ đề: Khái niệm về hệ cơ sở dữ liệu; hệ quản trị cơ sở dữ liệu Microsoft Access; hệ cơ sở dữ liệu quan hệ; kiến trúc và bảo mật các hệ cơ sở dữ liệu (Ho et al., 2012).

Cite this article as: Phan Tan Quoc, Phan Thi Kim Loan, & Truong Tan Khoa (2021). A survey on algorithms in computer science for the new general education program. *Ho Chi Minh City University of Education Journal of Science, 18*(2), 331-341.

1.2. Chương trình môn Tin học thuộc Chương trình giáo dục phổ thông mới

Trong chương trình giáo dục phổ thông (viết tắt CTGDPT) mới, năng lực tin học được đặc biệt chú trọng; ngoài năng lực tin học cốt lõi (trong đó có chủ đề Lập trình cơ bản ở lớp 10 và Kĩ thuật lập trình ở lớp 11), CTGDPT mới thể hiện sự phân hóa sâu về định hướng nghề nghiệp; mỗi học sinh có thể tự chọn các chuyên đề học tập (viết tắt CĐHT) theo một trong hai định hướng *Tin học ứng dụng* (viết tắt THƯD) hoặc *Khoa học máy tính* (viết tắt KHMT) (Ministry of Education and Training, 2018; Ministry of Education and Training, 2019).

1.3. So sánh chương trình môn Tin học thuộc CTGDPT hiện tại và CTGDPT mới

Trong CTGDPT mới, vai trò của môn Tin học THPT có nhiều thay đổi: Tin học là môn học được lựa chọn theo nguyện vọng và định hướng nghề nghiệp của học sinh, phân hóa theo hai định hướng THƯD và KHMT (trong CTGDPT hiện tại không phân hóa).

Trong CTGDPT mới, bên cạnh nội dung giáo dục cốt lõi (70 tiết/lớp/năm học), học sinh có thể chọn học một số CĐHT (35 tiết/lớp/năm) tùy theo sở thích, nhu cầu và định hướng nghề nghiệp. So với CTGDPT hiện tại, nếu một học sinh ở CTGDPT mới chọn môn Tin học (học cả nội dung cốt lõi và các CĐHT) thì số tiết mỗi học sinh sẽ học là 315 tiết so với 175 tiết ở CTGDPT hiện tại; tức bằng 180% số tiết so với CTGDPT hiện tại (Ministry of Education and Training, 2018; Ho et al., 2014). Trong CTGDPT mới, môn Tin học được xem bình đẳng như các môn học tự chọn khác như Vật lí, Hóa học, Sinh học, Lịch sử, Địa lí... và cũng là môn thi tuyển sinh đại học sau này. Các CĐHT theo định hướng KHMT tập trung vào các nội dung như: thuật toán, cấu trúc dữ liệu, ngôn ngữ lập trình. Trong CTGDPT mới, không quy định cụ thể về ngôn ngữ lập trình sẽ giảng dạy.

Bảng 1. *Tính tự chọn/bắt buộc của môn Tin học trong CTGDPT tổng thể*

Lớp	CTGDPT hiện tại	CTGDPT mới	
		Chương trình cốt lõi	Chuyên đề học tập
Lớp 10	Bắt buộc	Tự chọn	Tự chọn
Lớp 11	Bắt buộc	Tự chọn	Tự chọn
Lớp 12	Bắt buộc	Tự chọn	Tự chọn

Bảng 2. *Kiến thức về lập trình ở mỗi khối lớp*

Lớp	CTGDPT hiện tại	CTGDPT mới	
		Chương trình cốt lõi	Chuyên đề học tập
Lớp 10	Không	Lập trình cơ bản	3 CĐHT
Lớp 11	Lập trình cơ bản	Kĩ thuật lập trình	3 CĐHT
Lớp 12	Không	Không	3 CĐHT
Ngôn ngữ lập trình	Pascal	Java/Python/C#/…	Java/Python/C#/…

Bảng 3. *Thời lượng giảng dạy môn Tin học ở mỗi khối lớp*

Lớp	CTGDPT hiện tại	CTGDPT mới	
		Chương trình cốt lõi	Chuyên đề học tập
Lớp 10	70.0	70	35
Lớp 11	52.5	70	35
Lớp 12	52.5	70	35
Cấp THPT	175 tiết	315 tiết	

2. Khảo sát các thuật toán thuộc nội dung các CĐHT định hướng KHMT môn Tin học thuộc CTGDPT mới

Phần này trình bày về thuật toán, thuật toán đệ quy, thuật toán chia để trị, thuật toán quay lui thuộc các CĐHT định hướng KHMT môn Tin học lớp 11 thuộc CTGDPT mới và một số thuật toán liên quan.

2.1. Thuật toán và độ phức tạp

Trong bài học *Bài toán và thuật toán* (Ho et al., 2014), các tác giả đã trình bày thuật toán bằng ngôn ngữ sơ đồ khối cho một số bài toán sau:

- Tìm giá trị lớn nhất của một dãy số nguyên;
- Kiểm tra tính nguyên tố của một số nguyên dương;
- Bài toán sắp xếp;
- Bài toán tìm kiếm.

Học sinh lớp 11 CTGDPT mới đã học các kiến thức Lập trình cơ bản, Kĩ thuật lập trình và đang học về các CĐHT định hướng KHMT, nên chúng tôi đề xuất một số bài toán có độ khó cao hơn cho nội dung liên quan đến thuật toán.

Với nội dung về thuật toán và độ phức tạp (algorithm and complexity), chúng tôi đề xuất các nội dung có thể tham khảo: Bài toán và thuật toán, độ phức tạp của thuật toán, một số lớp thuật toán, ngôn ngữ biểu diễn thuật toán, một số kĩ thuật phân tích thuật toán, đánh giá độ phức tạp của thuật toán bằng quy tắc cộng và quy tắc nhân, một số bài toán tổ hợp như bài toán đếm tổ hợp (counting problem), bài toán tồn tại tổ hợp (existence problem), bài toán liệt kê tổ hợp (enumeration problem), bài toán tối ưu tổ hợp (combinatorial optimization problem) (Nguyen, & Nguyen, 2009; Nguyen, 2013; Robert, & Kevin, 2011; Steven, & Felix, 2010; Jon, 2000).

Một số bài toán có thể làm ví dụ minh họa để thấy được sự quan trọng của độ phức tạp của thuật toán như:

- Bài toán cho dãy n số nguyên ($n \leq 10^6$), tìm dãy con liên tiếp có tổng lớn nhất; bài toán này có thể giải với độ phức tạp $O(n)$.
- Bài toán cho dãy n số nguyên ($n \leq 10^6$), chuyển đoạn dãy con gồm k phần tử đầu dãy về cuối dãy (yêu cầu không dùng mảng phụ); bài toán này có thể giải với độ phức tạp $O(n)$.
- Bài toán cho dãy n số nguyên ($n \leq 10^6$), tìm 3 số có tích lớn nhất; bài toán này có thể giải với độ phức tạp $O(nlogn)$.

Một số bài toán để rèn luyện cho nội dung về thuật toán như cho dãy gồm n số nguyên dương ($n \leq 10^6$):

- Tìm một dãy con liên tiếp tăng dài nhất; bài toán này có thể giải với độ phức tạp $O(n)$.
- Đếm xem trong dãy có bao nhiêu cặp số có tổng bằng M; bài toán này có thể giải với độ phức tạp $O(nlogn)$.
- Đếm xem trong dãy có bao nhiêu số có giá trị đôi một khác nhau.

Cũng có thể hướng dẫn mở rộng thêm bằng các thuật toán sắp xếp đơn giản, các bài toán về số học số nguyên lớn, các bài toán tính toán hình học đơn giản.

2.2. Thuật toán đệ quy

CĐHT Thực hành thiết kế thuật toán theo thuật toán đệ quy (recursive algorithm).

Nội dung: Khái niệm đệ quy, thực hành thiết kế thuật toán theo thuật toán đệ quy.

Yêu cầu cần đạt: Trình bày được tính đệ quy trong một vài định nghĩa sự vật, sự việc; xác định được phần cơ sở và phần đệ quy trong mô tả đệ quy; ứng dụng được thuật toán đệ quy trong thiết kế một vài thuật toán như: tính a^n; tính $n!$; tìm phần tử thứ n của dãy fibonaci; bài toán tháp Hà Nội; viết được chương trình sử dụng thuật toán đệ quy cho một vài bài toán đơn giản; nhận biết được tính ưu việt của thuật toán đệ quy trong định nghĩa sự vật, mô tả và thiết kế thuật toán (Ministry of Education and Training, 2018).

Với yêu cầu cần đạt và nội dung thuật toán đệ quy như trên, chúng tôi đề xuất các nội dung có thể tham khảo: Định nghĩa đệ quy, định nghĩa thuật toán đệ quy, đệ quy và quy nạp toán học, hàm đệ quy, vấn đề khử đệ quy, sơ đồ thuật toán đệ quy, phân tích thuật toán đệ quy, chứng minh tính đúng đắn của thuật toán đệ quy, đệ quy có nhớ (Nguyen, 2013; Le, 2012; Pham, & Do, 2018; Huynh, Phan, & Nguyen, 2016).

Thuật toán 1. Sơ đồ thuật toán đệ quy

```
1.  void Recursive(Input)
2.  {
3.      if (kích thước của Input là nhỏ nhất)
4.          Thực hiện Bước cơ sở;
5.      else
6.      {
7.          Recursive(Input với kích thước nhỏ hơn); /* Bước đệ quy */
8.          /*Có thể có thêm những lệnh gọi đệ quy */
9.          Tổ hợp lời giải của các bài toán con để thu được Lờigiải;
10.         return Lờigiải;
11.     }
12. }
```

Một số bài toán điển hình có thể ví dụ minh họa cho thuật toán đệ quy:

- Bài toán tính giai thừa;
- Bài toán tính số hạng của dãy fibonacci;

- Bài toán tính hệ số của nhị thức;
- Bài toán tính số hạng hai dãy số có quan hệ hỗ tương;
- Bài toán tính số hạng của một dãy số dạng đệ quy phi tuyến;
- Bài toán tháp Hà Nội;
- Rèn luyện về kĩ thuật đệ quy có nhớ qua một số bài toán như: tính số hạng của dãy fibonacci, tính hệ số của nhị thức.

Một số bài toán để rèn luyện cho thuật toán đệ quy như:

- Tìm ước số chung lớn nhất của hai số tự nhiên;
- Chuyển đổi một số trong hệ đếm thập phân thành số trong hệ đếm cơ số *b*;
- Tính x^n;
- Một số bài toán dạng đệ quy nhị phân, đệ quy hỗ tương, đệ quy phi tuyến;

Các thuật toán chia để trị, thuật toán quay lui thường được mô tả dưới dạng đệ quy; đây là các thuật toán có một vai trò quan trọng trong lĩnh vực thiết kế thuật toán.

2.3. Thuật toán chia để trị

CĐHT Thực hành thiết kế thuật toán theo thuật toán chia để trị (divide and conquer algorithm).

Nội dung: Thực hành thiết kế thuật toán theo thuật toán chia để trị.

Yêu cầu cần đạt: Giải thích được sơ lược về thuật toán chia để trị; nêu được ý tưởng thiết kế thuật toán theo thuật toán chia để trị, giải thích được mối liên hệ giữa thiết kế thuật toán theo thuật toán chia để trị và đệ quy, nêu được ví dụ minh hoạ; viết được chương trình đơn giản có sử dụng thuật chia toán để trị (Ministry of Education and Training, 2018).

Với yêu cầu cần đạt và nội dung của thuật toán chia để trị như trên, chúng tôi đề xuất các nội dung có thể tham khảo: Ý tưởng thuật toán chia để trị, sơ đồ thuật toán chia để trị.

Để giải quyết một vấn đề bài toán theo thuật toán chia để trị, ta chia bài toán cần giải thành nhiều bài toán con có thể giải quyết một cách độc lập, giải quyết (trị) từng bài toán con này, sau đó từ các kết quả này, xây dựng (tổ hợp) kết quả cho bài toán cần giải ban đầu. Chia để trị là hướng tiếp cận bài toán từ trên xuống; tức nhìn vào bài toán lớn-chia nhỏ ra thành các bài toán con-trị các bài toán con; viêc giải quyết các bài toán con này thường được thực hiện đệ quy (Nguyen, 2013; Do, 2015; Anany, 2012).

Thuật toán chia để trị gồm các thao tác sau:

Chia (divide): Chia bài toán cần giải thành một số bài toán con có thể giải quyết độc lập; các bài toán con có kích thước nhỏ hơn và cùng dạng với bài toán cần giải.

Trị (conquer): Giải các bài toán con; thường sử dụng thuật toán đệ quy; các bài toán con có kích thước đủ nhỏ sẽ được giải trực tiếp.

Tổ hợp (combine): Tổ hợp lời giải của các bài toán con; thu đươc lời giải của bài toán xuất phát.

Thuật toán 2. Sơ đồ thuật toán chia để trị

```
1.  void DivideConquer(A,x); // tìm nghiệm x của bài toán A
2.  {
3.   if (A đủ nhỏ)
4.          Solve(A); //A đủ nhỏ để có thể giải trực tiếp
5.   else
6.   {
7.          Phân A thành các bài toán con A1, A2,…, Am;
8.          for (i=1;i<=m;i++)
9.          DivideConquer(Ai, xi);
10.         Kết hợp các nghiệm xi (i=1,2,..,m) của các bài toán con Ai
                 để nhận được nghiệm của bài toán A;
11.  }
12. }
```

Một số bài toán điển hình có thể làm ví dụ minh họa cho thuật toán chia để trị: Bài toán tìm kiếm nhị phân; bài toán chuyển một đoạn dãy con từ đầu dãy về cuối dãy; bài toán tính x^n; bài toán sắp xếp với thuật toán trộn (Merge sort).

Một số bài toán để rèn luyện thuật toán chia để trị như: sắp xếp Quick sort; sắp xếp Heapsort; tìm cặp điểm gần nhất; tính tích hai ma trận.

2.4. Thuật toán quay lui

CĐHT Thực hành thiết kế thuật toán theo thuật toán duyệt, thuật toán quay lui (backtracking algorithm).

Nội dung: Thuật toán duyệt, thuật toán quay lui.

Yêu cầu cần đạt: Nêu được ý tưởng của thuật toán duyệt và ví dụ minh hoạ (như tìm phần tử lớn nhất hoặc nhỏ nhất trong một dãy số, tìm một số trong một dãy số...); nêu được ý tưởng của thuật toán quay lui và nêu được ví dụ minh hoạ (như in các xâu nhị phân độ dài *n*, tìm tất cả các hoán vị của *n* phần tử...); nhận ra được mối liên quan giữa thiết kế thuật toán theo thuật toán quay lui và thuật toán đệ quy; viết được chương trình đơn giản có sử dụng thuật toán duyệt; viết được chương trình đơn giản có sử dụng thuật toán quay lui (Ministry of Education and Training, 2018).

Với yêu cầu cần đạt và nội dung của thuật toán quay lui và thuật toán duyệt như trên, chúng tôi đề xuất các nội dung có thể tham khảo: bài toán liệt kê, thuật toán quay lui; một số bài toán áp dụng thuật toán duyệt được liệt kê trong CTGDPT mới là khá đơn giản.

Thuật toán quay lui là một trong những thuật toán cơ bản và quan trọng được áp dụng để giải quyết nhiều vấn đề bài toán. Tất cả các bài toán liệt kê tổ hợp cơ bản đều có thể phát biểu dưới dạng bài toán liệt kê (Nguyen, 2013; Do, 2015; Anany, 2012).

Thuật toán 3. Sơ đồ thuật toán quay lui

```
1.  void Bactrack(int i)
2.  {
3.   for (int j=1;j<=n;j++)
4.   if (<chấp nhận j>)
5.   {
6.          <Xác định xi theo j>;
7.          if (i==n)
8.                 <Ghi nhận một cấu hình>
9.          else
10.                Bactrack(i+1);
11.  }
12.  }//Lệnh gọi để thực hiện thuật toán quay lui là: Bactrack(1);
```

Một số bài toán điển hình có thể làm ví dụ minh họa cho thuật toán quay lui:

- Liệt kê các dãy nhị phân độ dài n;
- Liệt kê các dãy hoán vị của $\{1,2,\ldots,n\}$;
- Liệt kê các tổ hợp chặp m của $\{1,2,\ldots,n\}$;
- Liệt kê các chỉnh hợp lặp chặp k của $\{1,2,\ldots,n\}$;
- Bài toán xếp n quân Hậu (the n-Queens problem).

Một số bài toán để rèn luyện thuật toán quay lui như:

- Bài toán người đi du lịch (traveling saleman problem);
- Bài toán cho dãy số, tìm các tập con có tổng bằng s cho trước;
- Bài toán cho một số nguyên dương n ($n \leq 30$), hãy tìm tất cả các cách phân tích số n thành tổng của các số nguyên dương, các cách phân tích là hoán vị của nhau chỉ tính là một cách;
- Bài toán tìm một cách chia n số trên thành hai phần sao cho tổng các phần tử của mỗi phần chênh lệch nhau là ít nhất;
- Bài toán cho ma trận vuông n x n, các phần tử là các số nguyên, hãy chọn ra n số sao cho n số này không cùng dòng, không cùng cột và có tổng lớn nhất;
- Bài toán cho dãy gồm n số nguyên, tìm các bộ k số có tổng bằng m cho trước (k, m là các số nguyên dương);
- Bài toán tìm phương án trả tiền từ cây ATM: một cây ATM hiện có n ($n \leq 200$) tờ tiền có mệnh giá $t_1,t_2,\ldots,t_n$, tìm một phương án trả (liệt kê các giá trị tờ tiền) có tổng giá trị là S;
- Bài toán tìm đường đi trong mê cung;
- Bài toán Mã đi tuần;
- Một số bài toán tối ưu rời rạc: bài toán đóng thùng; bài toán phân công (Nguyen & Nguyen, 2009).

2.5. Bàn luận thêm về một số thuật toán liên quan

Tiếp theo chúng tôi bàn luận về thuật toán nhánh cận, thuật toán sinh, thuật toán quy hoạch động và thuật toán tham lam. Trong đó thuật toán nhánh cận, thuật toán sinh liên quan đến thuật toán quay lui. Thuật toán quy hoạch động có hướng tiếp cận dạng "quy hoạch" bài toán; tức quá trình đưa bài toán ban đầu về một dạng nào đó có thể áp dụng thuật toán này để giải (nhìn qua thuật toán quy hoạch động có nét tương tự như thuật toán chia để trị về hướng tiếp cận, sự khác nhau của hai thuật toán này chúng tôi sẽ trình bày rõ ở phần sau). Thuật toán tham lam thường được sử dụng để giải quyết các bài toán tối ưu có kích thước lớn thay vì sử dụng các thuật toán quay lui hoặc thuật toán nhánh cận hoặc thuật toán quy hoạch động.

❖ ***Thuật toán nhánh cận***

Thuật toán nhánh cận (branch and bound algorithm) là một cải tiến được biết đến rộng rãi của thuật toán quay lui. Thuật toán nhánh cận nhằm đưa ra các điều kiện đánh giá cận hợp lí cho phép cắt bỏ các nhánh mà chắc chắn các nhánh đó không dẫn đến lời giải tối ưu. Thuật toán nhánh cận cho phép giải được bài toán có kích thước lớn hơn so với việc sử dụng giải bằng thuật toán quay lui.

Một số bài toán điển hình có thể làm ví dụ minh họa cho thuật toán nhánh cận như (Nguyen & Nguyen, 2009; Anany, 2012): Bài toán người đi du lịch; bài toán cái túi.

❖ ***Thuật toán sinh***

Thuật toán sinh (generate algorithm) là một trong những thuật toán quan trọng để giải bài toán liệt kê tổ hợp; tùy theo vấn đề bài toán mà có cách sinh phù hợp; làm giảm đáng kể cấu hình tổ hợp để tối ưu về thời gian tính.

Một số bài toán điển hình có thể làm ví dụ minh họa cho thuật toán sinh (Nguyen, & Nguyen, 2009; Do, 2015; Pham, & Do, 2018):

- Sinh các dãy nhị phân độ dài n;
- Sinh các hoán vị của tập n phần tử;
- Sinh các tập con m phần tử của tập n phần tử.

❖ ***Thuật toán quy hoạch động***

Như phần trước đã đề cập, thuật toán chia để trị chia bài toán lớn cần giải thành nhiều bài toán con có thể giải quyết *độc lập*. Trong thuật toán quy hoạch động (dynamic programming algorithm), việc thực hiện nguyên lí này được đẩy đến cực độ: Khi không biết chắc cần giải quyết bài toán con nào, chúng ta giải quyết tất cả các bài toán con và lưu trữ kết quả những lời giải này với mục đích sử dụng lại chúng theo một sự phối hợp nào đó để giải quyết các bài toán tổng quát hơn.

So sánh với thuật toán chia để trị ta thấy, thuật toán chia để trị tiếp cận bài toán từ trên xuống (top-down): nhìn ngay vào vấn đề lớn, chia nhỏ thành các bài toán con độc lập, trị các bài toán con; còn thuật toán quy hoạch động tiếp cận bài toán từ dưới lên (bottom-up): bắt đầu từ các trường hợp đơn giản, xây dựng dần lên để giải quyết bài toán ở cấp độ

lớn hơn, đến bài toán cần giải (Nguyen, & Nguyen, 2009; Do, 2015; Pham, & Do, 2018; Anany, 2012).

Khi sử dụng thuật toán quy hoạch động, có thể gặp hai khó khăn sau đây: *thứ nhất*, không phải lúc nào sự kết hợp lời giải của các bài toán con cũng cho ra lời giải của bài toán lớn hơn; *thứ hai*, số lượng các bài toán con cần giải quyết và lưu trữ có thể rất lớn, không thể chấp nhận được (Robert, & Kevin, 2011; Nguyen, & Nguyen, 2009; Le, 2012).

Một số bài toán điển hình có thể làm ví dụ minh họa cho thuật toán quy hoạch động:

- Bài toán tìm dãy con tăng dài nhất;
- Bài toán cái túi;
- Bài toán tìm đường di chuyển trên bảng hoặc trên tam giác số sao cho thỏa điều kiện tối ưu nào đó;
- Bài toán tìm dãy con chung dài nhất trong hai dãy số.

Một số bài tập để rèn luyện thuật toán quy hoạch động như:

- Bài toán cái túi (bài toán cái túi có nhiều biến thể thường được gọi là bài toán cái túi 1, bài toán cái túi 2, bài toán cái túi 3);
- Bài toán chia kẹo;
- Bài toán tìm dãy con gồm nhiều phần tử nhất sao cho tổng các phần tử chia hết cho k;
- Bài toán cho một đa giác lồi n đỉnh thành n–2 tam giác bằng các đường chéo không cắt nhau sao tổng các đường chéo là ngắn nhất.

❖ ***Thuật toán tham lam***

Thuật toán tham lam (greedy algorithm) dựa vào sự đánh giá tối ưu cục bộ địa phương (local optimum) để đưa ra quyết định tức thì tại mỗi bước lựa chọn, với hy vọng cuối cùng sẽ tìm được lời giải tối ưu toàn cục (global optimum). Thuật toán tham lam có thể tìm được lời giải tối ưu mà cũng có thể không. Trong trường hợp thuật toán tham lam không tìm được lời giải tối ưu, chúng ta thường thu được một lời giải có chất lượng chấp nhận được trong khoảng thời gian chấp nhận được để lời giải đó là có ý nghĩa (Nguyen, & Nguyen, 2009; Do, 2015; Anany, 2012).

Một số bài toán điển hình có thể làm ví dụ minh họa và bài tập rèn luyện cho thuật toán tham lam như: Bài toán người đi du lịch; bài toán tô màu đồ thị; bài toán tìm tập các đoạn thẳng không giao nhau; bài toán lập lịch/bài toán phân việc; bài toán tìm phương án trả tiền từ cây ATM; bài toán bố trí phòng họp; bài toán tìm cách nối các địa điểm sao cho tổng các đoạn nối là nhỏ nhất (mô hình của bài toán tìm cây khung nhỏ nhất); bài toán chọn k đối tượng sao cho số k là lớn nhất và các đối tượng được chọn không giao nhau; bài toán cái túi với trọng lượng, giá trị của các đồ vật là số thực hoặc khi kích thước bài toán đủ lớn.

Để việc triển khai các CĐHT có hiệu quả, các giáo viên cần dựa vào nội dung và yêu cầu cần đạt của mỗi CĐHT để xác định rõ mục tiêu cho từng bài học cụ thể; các ví dụ và các bài tập đưa ra nhằm hướng tới đáp ứng các mục tiêu đó.

3. Kết luận

Bài báo này đã khảo sát các thuật toán thuộc các chuyên đề học tập định hướng Khoa học máy tính lớp 11 thuộc chương trình giáo dục phổ thông mới bao gồm thuật toán đệ quy, thuật toán chia để trị, thuật toán quay lui. Bên cạnh đó, bài báo cũng đã bàn luận thêm về các thuật toán nhánh cận, thuật toán sinh, thuật toán quy hoạch động, thuật tham lam. Với mỗi chuyên đề, chúng tôi đã chi tiết hóa nội dung bằng cách đề xuất nội dung lí thuyết liên quan, danh sách các ví dụ minh họa điển hình và danh sách các bài toán rèn luyện. Bài báo này có thể làm tài liệu hỗ trợ giảng dạy và học tập các chuyên đề học tập định hướng Khoa học máy tính lớp 11 thuộc chương trình giáo dục phổ thông mới.

❖ ***Tuyên bố về quyền lợi:*** *Các tác giả xác nhận hoàn toàn không có xung đột về quyền lợi.*

❖ ***Lời cảm ơn:*** *Bài báo này nhận được sự tài trợ từ Đề tài khoa học và công nghệ của Trường Đại học Sài Gòn: Xây dựng tài liệu hỗ trợ việc giảng dạy và học tập môn Tin học lớp 11 thuộc chương trình giáo dục phổ thông mới, mã số: CS2019-29.*

TÀI LIỆU THAM KHẢO

Anany, L. (2012). *Introduction to The design and Analysis of Algorithms*. Addison-Wesley, 565pages.

Do, P. T. (2015). *Competitive Programming*. Hanoi University of Science and Technology.

Ho, S. D., Tran, D. H., Nguyen, X.M., Nguyen, D. N., Nguyen, T. T., & Ngo, A. T. (2014). *Informatics 10th grade*. Vietnam Education Publishing House, 177 pages.

Ho, S. D., Ho, C. H., Tran, D. H., Nguyen, D. N., Nguyen, T. T., & Ngo, A. T. (2014). *Informatics 11th grade*. Vietnam Education Publishing House, 144 pages.

Ho, S. D., Ho, C. H., Tran, D. H., Nguyen, D. N., Nguyen, T. T., & Ngo, A. T. *Informatics 12th grade*. Vietnam Education Publishing House, 136 pages.

Huynh, M. T., Phan, T. Q., & Nguyen, N. D. (2016). *Programming techniques*. Vietnam National University Publishing House, Ho Chi Minh City, 277 trang.

Jon, B. (2000). *Programming Pearls (Second edition)*. Addison-Wesley, Inc. 283 pages.

Le, M. H. (2012). *Lecture on some advanced informatics topics*. Hanoi National University of Education, 119 pages.

Ministry of Education and Training (2018). *General education program*. 32/2018/Circular - Ministry of Education and Training, 1555 pages.

Ministry of Education and Training (2019). *Conference documents deployment general education program*. Ministry of Education and Training, 1/2019, 90 pages.

Nguyen, D. N. (2013). *Data Structures and Algorithms*. Hanoi University of Science and Technology Publishing House, 283 pages.

Nguyen, D. N, & Nguyen, T. T. (2009). *Discrete math (sixth edition)*. Vietnam National University Publishing House, Hanoi, 290 pages.

Pham, Q. D., & Do, P. T. (2018). *Introduction to Data Structures and Algorithms*. Hanoi University of Science and Technology.

Robert, S., & Kevin, W. (2011). *Algorithms (fourth edition)*. Addison wesley, 955 pages.

Steven, H., & Felix, H. (2010). *Competitive Programming in National University of Singapore*, National University of Singapore, 140 pages.

Tran, H. L. (2018). *Applications online in teaching the program for children*, University of Science and Technology, The University of Danang.

Vuong, Q. H. (2019), *The role of research in Vietnamese education in era 4.0*, Phenikaa University, Hanoi.

A SURVEY ON ALGORITHMS IN COMPUTER SCIENCE FOR THE NEW GENERAL EDUCATION PROGRAM

Phan Tan Quoc*[*], Phan Thi Kim Loan, Truong Tan Khoa

Saigon University, Việt Nam

[*] *Corresponding author: Phan Tan Quoc – Email: quocpt@sgu.edu.vn*

Received: March 20, 2020; Revised: April 29, 2020; Accepted: February 25, 2021

ABSTRACT

The new general education program will be deployed from the 2022-2023 for high school level. The Informatics curriculum of the new program is designed with career-oriented core contents and learning topics. These core topics and learning topics are currently presented very briefly on the requirements to be achieved. The learning topics are designed in two orientations: Applied Informatics and Computer Science. The former focuses on the use of essential common software to improve work efficiency to create opportunities for students to make practical digital products for learning and life. The later focuses on developing computer thinking, problem analysis capacity, data type selection and algorithm design for students. In this paper, we surveyed and listed the contents of computer science-oriented topics of 11th grade Informatics program, including recursive, divide and conquer algorithm and backtracking algorithms. This paper aims to give an overview of the 11th grade Informatics algorithms as a source of references for high school teachers.

***Keywords*:** Backtracking algorithm; Branch and bound algorithm; Divide and conquer algorithm; Dynamic programming algorithm; Generate algorithm; Greedy algorithm; Recursive algorithm