Lý thuyết Cấu trúc dữ liệu và giải thuật

# I. Thuật toán sinh

Thuật toán sinh được dùng để giải lớp các bài toán thỏa mãn hai điều kiện:

•Xác định được một thứ tự trên tập các cấu hình cần liệt kê của bài toán. Biết được cấu hình đầu tiên, biết được cấu hình cuối cùng.

•Từ một cấu hình cuối cùng, ta xây dựng được thuật toán sinh ra cấu hình đứng ngay sau nó theo thứ tự.

# II. Thuật toán quay lui (Back track)

Giả sử ta cần xác định bộ X =(x1 , x2 ,..,xn ) thỏa mãn một số ràng buộc nào đó. Ứng với mỗi thành phần xi ta có ni khả năng cần lựa chọn. Ứng với mỗi khả năng j∈ni dành cho thành phần xi ta cần thực hiện:

• Kiểm tra xem khả năng j có được chấp thuận cho thành phần xi hay không? Nếu khả năng j được chấp thuận thì nếu i là thành phần cuối cùng (i=n) ta ghi nhận nghiệm của bài toán. Nếu i chưa phải cuối cùng ta xác định thành phần thứ i +1.

• Nếu không có khả năng j nào được chấp thuận cho thành phần xi thì ta quay lại bước trước đó (i-1) để thử lại các khả năng khác.

# III. Thuật toán tham lam

Giới thiệu thuật toán Thuật toán tham lam (Greedy Algorithm): là thuật toán xem xét quá trình giải quyết bài toán bằng việc tạo nên lựa chọn tối ưu cục bộ tại mỗi bước thực hiện với mong muốn tìm ra lựa chọn tối ưu toàn cục. Giải thuật tham lam thường không mang tính tổng quát. Tuy vậy, bằng việc xem xét các lựa chọn tối ưu cục bộ cho ta lời giải gần đúng với phương án tối ưu toàn cục cũng là giải pháp tốt trong thời gian chấp nhận được.

# VI. Thuật toán chia để trị

Thuật toán chia để trị (Devide and Conquer): dùng để giải lớp các bài toán có thể thực hiện được ba bước:

1. Devide (Chia). Chia bài toán lớn thành những bài toán con có cùng kiểu với bài toán lớn.

2. Conquer (Trị). Giải các bài toán con. Thông thường các bài toán con chỉ khác nhau về dữ liệu vào nên ta có thể thực hiện bằng một thủ tục đệ qui.

3. Combine (Tổng hợp). Tổng hợp lại kết quả của các bài toán con để nhận được kết quả của bài toán lớn

# V. Thuật toán qui hoạch động (Dynamic Programming)

Phương pháp qui hoạch động dùng để giải lớp các bài toán thỏa mãn những điều kiện sau:

• Bài toán lớn cần giải có thể phân rã được thành nhiều bài toán con. Trong đó, sự phối hợp lời giải của các bài toán con cho ta lời giải của bài toán lớn. Bài toán con có lời giải đơn giản được gọi là cơ sở của qui hoạch động. Công thức phối hợp nghiệm của các bài toán con để có nghiệm của bài toán lớn được gọi là công thức truy hồi của qui hoạch động.

• Phải có đủ không gian vật lý lưu trữ lời giải các bài toán con (Bảng phương án của qui hoạch động). Vì qui hoạch động đi giải quyết tất cả các bài toán con, do vậy nếu ta không lưu trữ được lời giải các bài toán con thì không thể phối hợp được lời giải giữa các bài toán con.

• Quá trình giải quyết từ bài toán cơ sở (bài toán con) để tìm ra lời giải bài toán lớn phải được thực hiện sau hữu hạn bước dựa trên bảng phương án của qui hoạch động.

# IV. Danh sách liên kết:

Có 4 loại danh sách liên kết đơn: danh sách liên kết đơn, danh sách liên kết đơn vòng, danh sách lien kết kép, danh sách lien kết kép vòng.

## 1. Danh sách liên kết đơn:

* Định nghĩa: Tập các Node thông tin được tổ chức rời rạc trong bộ nhớ. Trong đó mỗi node gồm 2 thành phần:
  + Thành phần dữ liệu: dùng để lưu thông tin của node.
  + Thành phần con trỏ: dùng để trỏ đến node dữ liệu tiếp theo.
* Các thao tác trên danh sách danh sách đơn:
  + Khởi tạo danh sách liên kết đơn.
  + Cấp phát miền nhớ cho một node.
  + Thêm node vào đầu bên trái, phải.
  + Thêm node vào giữa danh sách.
  + Loại node cuối bên trái, phải
  + Loại node ở giữa
  + Duyệt thông tin danh sách.
  + Tìm node trên danh sách
* Ứng dụng danh sách liên kết đơn:
  + Xây dựng các lược đồ quản lý bộ nhớ
  + Biểu diễn ngăn xếp
  + Biểu diễn hàng đợi
* Biểu diễn cây biểu diễn đồ thị
* Biểu diễn tính toán

## 2. Danh sách liên kết đơn vòng

* Định nghĩa: Là danh sách lien kết đơn trong đó tất cả các node liên kết với nhau thành một vòng tròn. Không có con trỏ NULL ở node cuối cùng mà được lien kết với node đầu tiên.
* Tính chất liên kết vòng:
  + Mọi node đều là node bắt đầu.
  + Dễ dàng trong việc cài đặt hàng đợi
  + Dễ dàng phát triển các ứng dụng thực hiên vòng quanh danh sách.
* Các thao tác trên danh sách liên kết vòng

## 3. Danh sách liên kết kép:

* Định nghĩa: Tập các node được tổ chức rời rạc trong bộ nhớ. Trong đó, mỗi node có 3 thành phần:
  + Thành phần dữ liệu: dùng để lưu trữ thông tin node.
  + Thành phần con trỏ prev: dùng để trỏ đến node dữ liệu trước nó.
  + Thành phần con trỏ next: dùng để trỏ đến node dữ liệu sau nó.
* Các thao tác trên danh sách liên kết kép:

## 4. Danh sách liên kết kép vòng

* Định nghĩa: Là danh sách liên kết kép trong đó node cuối được liên kết với ndoe đầu.

# VII. Cây nhị phân:

## 1. Định nghĩa:

* Tập hợp hữu hạn các node có cùng kiểu dữ liệu được phân thành 3 tập:
  + Tập thứ nhất có thể rỗng hoặc chỉ có 1 node gọi là node gốc (root)
  + Hai tập con còn lại tự hình thành 2 cây con. Cây con trái và cây con phải của node gốc.

## 2. Một số khái niệm:

* Node gốc (root): Node đầu tiên định hình cây
* Node cha: Node A là node cha của node B nếu B hoặc là node con trái, hoặc là node con phải của A.
* Node lá: Node không có node con trái, không có node con phải.
* Node trung gian: Node hoặc có node con trái hoặc có node con phải hoặc cả hai.
* Bậc của node: Số cây con tối đa của node
* Mức của node: mức node gốc có bậc là 0, mức của các node khác trên cây bằng mức của node cha cộng 1.
* Chiều sâu của cây: mức lớn nhất của node lá trong cây.

## 3. Các loại cây nhị phân:

* **Cây lệch trái:** cây chỉ có các node con bên trái
* **Cây lệch phải:** cây chỉ có các node con bên phải.
* **Cây nhị phân đúng:** Node gốc và tất cả các node trung gian có đúng hai node con
* **Cây nhị phân gần đầy:** Cây nhị phân gần đầy có chiều sâu d là cây nhị phân thảo mãn:
  + Tất cả các node con có mức không nhỏ hơn d-1 dều có 2 node con.
  + Các node ở mức d đầy từ trái qua phải.
* **Cây nhị phân hoàn toàn cân bằng:** Cây nhị phân có số node thuộc nhánh cây con trái và số node thuộc nhánh cây con phải chênh lệnh nhau không quá 1.
* **Cây nhị phân tìm kiếm:** Cây nhị phân thỏa mãn điều kiện:
  + Hoặc là rỗng hoặc là có 1 node gốc.
  + Mỗi node gốc có tối đa hai cây con. Nội dung node gốc lớn hơn nội dung node con bên trái và nhỏ hơn nội dung ndoe con bên phải.
  + Hai cây con bên trái và bên phải cũng tạo thành hai cây tìm kiếm.
* Cây nhị phân tìm kiếm hoàn toàn cân bằng: Cây nhị phân tìm kiếm có chiều sâu cây con bên trái và chiều sâu cây con bên phải chênh lệch nhau không quá 1.

4. Biểu diễn cây nhị phân:

* + Biểu diễn liên tục: sử dụng mảng trong đó:
    - Node gốc lưu trữ ở vị trí 0.
    - Nếu node cha lưu trữ ở vị trí p thì node con trái sẽ được lưu ở vị trí 2p+1, và node con phải sẽ được lưu trữ ở vị trí 2p+2.
  + Biểu diễn rời rạc: Sử dụng danh sách liên kết đơn.

## 5. Các thao tác trên cây nhị phân:

* Tạo node gốc cho cây.
* Thêm vào node lá bên trái hoặc phải của node p
* Loại bỏ node là bên trái hoặc phải của node p
* Loại bỏ cây
* Tìm kiếm node trên cây.
* Duyệt cây theo thứ tự trước (root – left – right)
* Duyệt cây theo thứ tự giữa (left – root – right)
* Duyệt theo thứ tự sau (left – right – root)
* Duyệt cây theo mức (Level)

## 6. B-tree:

* Định nghĩa: B-tree bậc M là cây nhị phân tìm kiếm bậc M có tính chất:
  + Mỗi node (ngoại trừ node gốc) có ít nhất M/2 node con.
  + Node gốc có ít nhất 2 node con.
  + Mọi node lá đều nằm trong cùng một mức.
  + Các khóa và cây con được sắp xếp theo cây tìm kiếm.

## 7. Cây đỏ đen:

* Định nghĩa: Là cây nhị phân tìm kiếm có tính chất:
  + Mọi node phải có tính chất đỏ đen.
  + Node gốc và các node luôn là đen.
  + Nếu một node là đỏ thì node con của nó là đen.
  + Đường đi từ node gốc đến node lá phải có cùng số lượng node đen.

# VIII. Đồ thi:

* Định nghĩa: Đồ thị G = <V, E> trong đó V là tập hợp hữu hạn được gọi là tập đỉnh, E là tập có thứ tự hoặc không có thứ tự các cặp đỉnh được gọi là tập cạnh.
* Đồ thị vô hướng: Đồ thị G = <V, E> được gọi là vô hướng nếu các cạnh thuộc E là các bộ không tính thứ tự các đỉnh trong V.
* Đơn đồ thị vô hướng: Đồ thị G = <V, E> là đơn đồ thị vô hướng nếu là đồ thị vô hướng và giữa hai đỉnh bất kì thuộc V có nhiều nhất 1 cạnh nối.
* Đơn đồ thị có hướng: Đồ thị G = <V, E> bao gồm V là tập các đỉnh, E là tập các cặp có thứ tự bao gồm hai phần tử của V gọi là cách cung.

# IX. Stack:

* Định nghĩa: Tập các node thông tin được tổ chức liên tục hoặc rời rạc nhau trong bộ nhớ và thực hiện theo cơ chế LIFO (Last In First Out) vào cuối ra trước.
* Biểu diễn ngăn xếp:
  + Biểu diễn liên tục: các phần tử dữ liệu của ngăn xếp được lưu trữ liên tục nhau trong bộ nhớ (Mảng)
  + Biểu diễn rời rạc: Các phần tử của ngăn xếp được lưu trữ rời rạc nhau trong bộ nhớ (danh sách liên kết đơn).
  + Các thao tác trên stack:
    - Kiểm tra tích rỗng của stack
    - Kiểm tra tính đầy của ngăn xếp
    - Đưa dữ liệu vào ngăn xếp: chỉ thực hiện khi ngăn xếp chưa tràn.
    - Lấy dữ liệu ra khỏi ngăn xếp: chỉ thực hiện khi ngăn xếp không rỗng.
  + Ứng dụng của stack:
    - Xây dựng các giải thuật đệ qui
    - Khử đệ qui
    - Biểu diễn tính toán
    - Duyệt cây, duyệt đồ thị

# X. Queue:

* Định nghĩa: Tập hợp các node thong tin đươc tổ chức lien tục hoặc rời rạc nhau trong bộ nhớ và thực hiện theo cơ chế FIFO (Fist In Fist Out)
* Biểu diễn hàng đợi: có 2 cách biểu diễn:
  + Biểu diễn liên tục: sử dụng mảng
  + Biểu diễn rời rạc sử dụng danh sách liên kết đơn.
* Các thao tác trên hàng đợi:
  + Kiểm tra tính rỗng của hàng đợi
  + Kiểm tra tính đầy của hàng đợi
  + Thao tác đưa dữ liệu vào hàng đợi
  + Thao tác lấy dữ liệu ra khỏi hàng đợi
* Các kiểu hàng đợi:
  + Hàng đợi tuyến tính
  + Hàng đợi vòng
  + Hàng đợi ưu tiên
* Ứng dụng của hàng đợi:
  + Biểu diễn tính toán
  + Duyệt cây
  + Xay dựng các thuật toán lập lịch

·