Kỹ thuật Giảm để trị (Decrease-and-Conquer)

Giới thiệu chung

Giảm với lượng không đổi (*decrease by a constant*)
Kích thước dữ liệu giảm một lượng không đổi sau mỗi bước lặp, thường là 1.

Ví dụ: Tính a^n.

$$f(n) = \begin{cases} f(n-1) \times a & n > 1 \\ a & n = 1 \end{cases}$$

Ví dụ: Sắp xếp chèn với dãy số $a_1, a_2, \dots, a_n$.

Giảm với tỉ lệ không đổi (*decrease by a constant factor*)
Kích thước dữ liệu giảm bởi tỉ lệ α (thường là $\frac{1}{2}$) không đổi sau mỗi bước lặp.

Ví dụ: Tìm kiếm nhị phân. Lời giải với dữ liệu kích thước n cũng vẫn là lời giải với dữ liệu kích thước $\frac{n}{2}$.

Ví dụ: Tính a^n.

$$a^n = \begin{cases} (a^{n/2})^2 & n \equiv_2 0 \wedge n \neq 0 \\ \left(a^{\lfloor n/2 \rfloor}\right)^2 \times a & n \equiv_2 1 \\ 1 & n = 0 \end{cases}$$

Ví dụ: Cách nhân của tá điền Nga

$$n \times m = \begin{cases} \dfrac{n}{2} \times 2m & n \equiv_2 0 \\ \left\lfloor \dfrac{n}{2} \right\rfloor \times 2m + m & n \equiv_2 1 \end{cases}$$

Giảm với lượng biến đổi (*variable size decrease*)
Kích thước của thể hiện bài toán giảm với lượng không xác định sau mỗi lượt lặp.

Ví dụ: Giải thuật Euclid để tìm USCLN của hai số a và b.

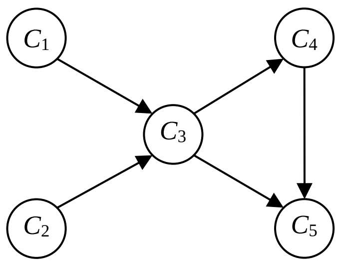
$$USCLN(a, b) = USCLN(b, a \% b)$$

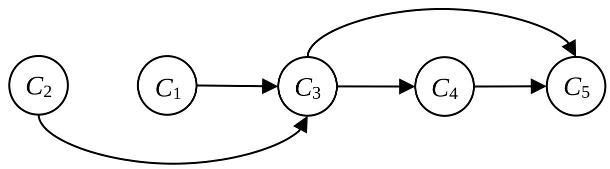
Ví dụ: Tìm một phần tử trên cây nhị phân tìm kiếm.

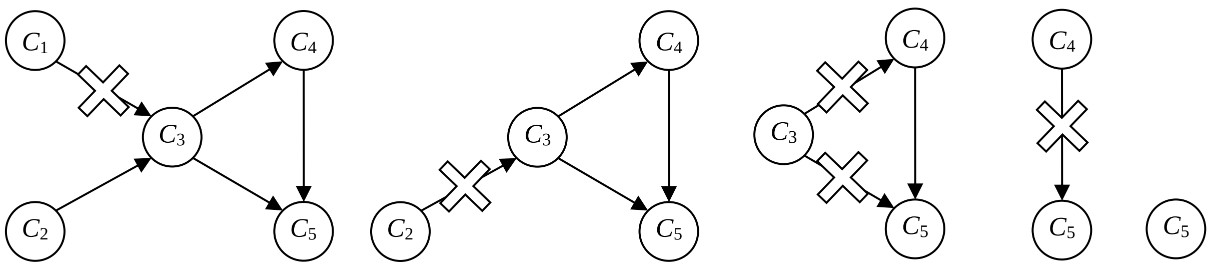
Kỹ thuật Giảm với lượng không đổi

Sắp xếp Topo

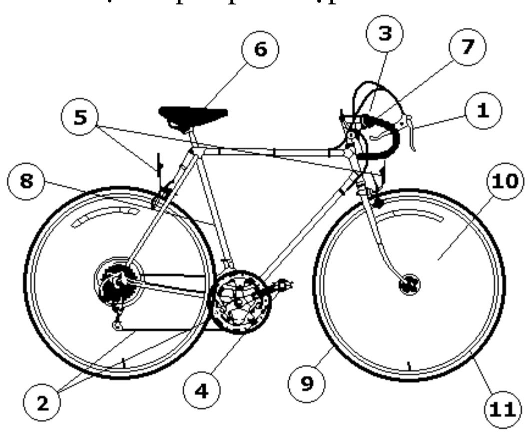
Ví dụ: Gọi $\{C_1, C_2, \ldots, C_5\}$ là tên của 5 môn học mà một sinh viên bắt buộc phải hoàn thành trong khóa học ngắn hạn nào đó.

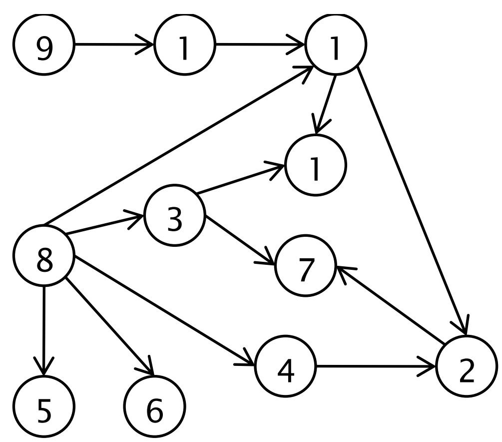




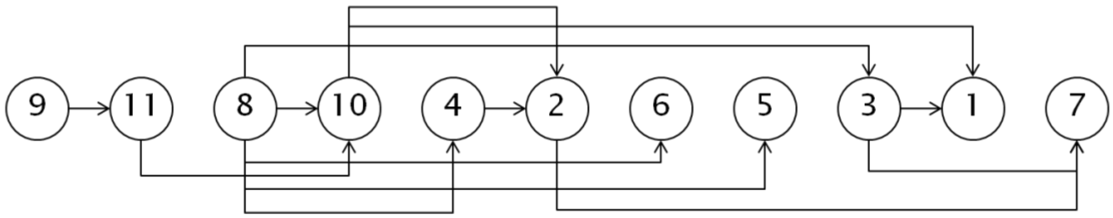


Ví dụ: Lắp ráp xe đạp





1. Lắp thắng vào tay lái
2. Gắn bộ truyền động
3. Gắn tay lái
4. Lắp bàn đạp và đĩa
5. Lắp đèn xe
6. Lắp yên xe
7. Lắp hộp điều chỉnh tốc độ
8. Lắp (tạo) khung xe
9. Lắp vỏ xe vào vành xe
10. Gắn bánh xe vào khung xe
11. Lắp vành xe vào bánh xe (đúc)



Giải thuật thô:

Bước 1: Duyệt mảng `indegree` và tìm ra những đỉnh có bậc vào bằng 0. Những đỉnh này sẽ được đưa lần lượt vào hàng chờ `Q`.

Bước 2: *Nếu* hàng chờ `Q` khác rỗng thì lấy đỉnh u ra khỏi hàng, in ra (lần lượt từ *trái sang phải*) và chuyển sang *Bước 3*;

Ngược lại: Dừng.

Bước 3: Với mỗi đỉnh v kề với đỉnh u, tiến hành giảm *bậc vào* của đỉnh v trong mảng `indegree` (tương đương việc loại các cạnh từ u đến v). Nếu phát hiện *bậc vào* của đỉnh v nào đó trở về 0 thì thêm vào hàng chờ `Q`. Quay về *Bước 2*.

Giải thuật:

```
TopologicalSort(Graph G) {
   indegree[1 .. |V|] = {0};
   Queue     Q = ∅;

   for (mỗi u ∈ V)
      for (mỗi đỉnh v: (u, v) ∈ E)
         indegree[v] ++;
   for (mỗi v ∈ V)
      if (indegree[v] == 0)
         enQueue(Q, v);

   while (!isEmpty(Q)) {
      Vertex u = deQueue(Q);
      Output(u);
      for (mỗi đỉnh v: (u, v) ∈ E)
         if( -- indegree[v] == 0 )
            enQueue(Q, v);
   }
}
```

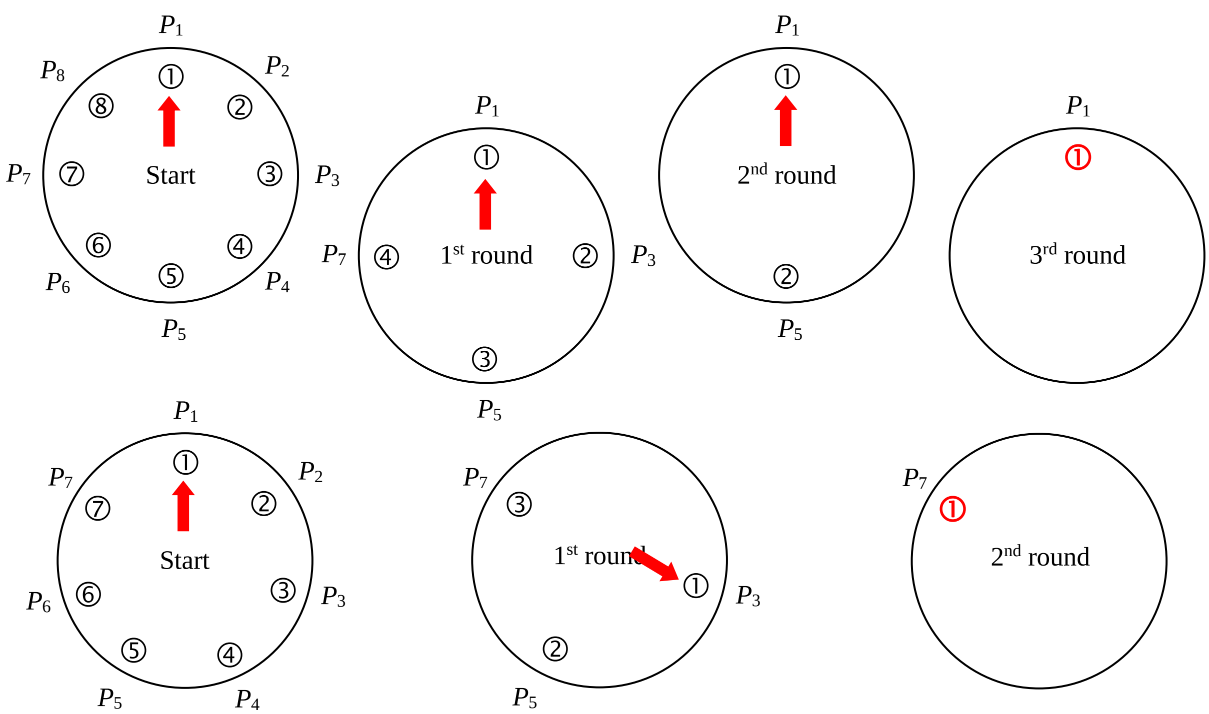
Đánh giá: Nếu đầu vào của giải thuật là danh sách kề thì chi phí của nó là $\Theta(|V| + |E|)$.

Kỹ thuật giảm với tỉ lệ không đổi

Bài toán Josephus

Phát biểu: Có n người đứng thành vòng tròn. Bắt đầu từ người có số thứ tự 1, người kế bên sẽ bị loại bỏ cho đến khi chỉ còn một người. *Hãy xác định số thứ tự $J(n)$ ban đầu của người sẽ trụ lại cuối cùng*.

Ví dụ: $J(8) = 1, J(7) = 7, J(6) = 5$.



Để xác định vị trí $J(n)$, chúng ta chia ra hai trường hợp:

- $n = 2h$:

$$J(2h) = 2J(h) - 1$$

- $n = 2h + 1$:

$$J(2h + 1) = 2J(h) + 1$$

Công thức tổng quát là:

$$J(2^k + i) = 2i + 1, i \in [0, 2^k - 1]$$

Kỹ thuật giảm với lượng biến đổi

Tìm USCLN(a, b) – Giải thuật Euclid

Phát biểu: Gọi $a, b \in \mathbb{Z}^+$. Tìm USCLN của hai số này.

Định lý (của) Lame: Gọi $a, b \in \mathbb{Z}^+$, với $a \geq b \geq 2$. Số lượng các phép chia nguyên của giải thuật Euclid để tìm ra $USCLN(a, b)$ không vượt quá 5 lần chiều dài của số nguyên b.

Giải thuật

```
gcd(a, b) {
   while (b) {
      t = b;
      b = a % b;
      a = t;
   }
   return   a;
}
```

Cây nhị phân tìm kiếm

Phát biểu: Xác định sự hiện diện của giá trị *k* trên cây nhị phân tìm kiếm.

Đánh giá:

- Trường hợp tốt nhất: $\Theta(\log n)$
- Trường hợp xấu nhất: $\Theta(n)$
- Trường hợp trung bình: $\Theta(\log n)$

Bài toán chọn (Selection problem)

Phát biểu: Tìm phần tử *nhỏ thứ k* trong dãy $S = \{s_1, s_2, \dots, s_n\}$, với $k \in [1, n]$.

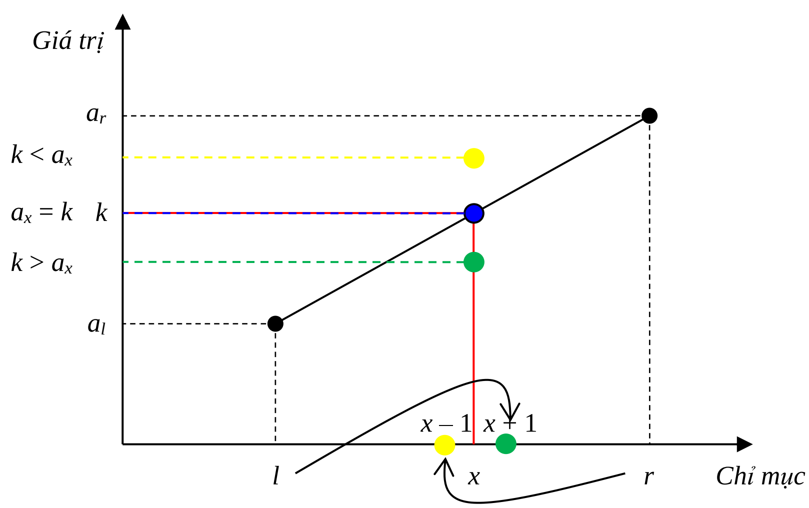
Đánh giá:

- Trường hợp tốt nhất: $\Theta(n)$
- Trường hợp xấu nhất: $O(n^2)$
- Trường hợp trung bình: $\Theta(n)$

Giải thuật

```
Selection(S[], lower, upper, k) {
   h = random(lower, upper);
   pos = Partition(S, lower, upper, h);    // S[pos] = pivot
   if (pos == k)
      return S[pos];
   if (pos > k)
      Selection(S, lower, pos - 1, k);
   if (pos < k)
      Selection(S, pos + 1, upper, k);
}
Partition(S, lower, upper, pos) {
   pivot = S[pos];
   S[lower] ⇆ S[pos];
   pos = lower;
   for (i = lower + 1; i ≤ upper; i++)
      if (pivot > S[i]) {
         pos++;
         S[i] ⇆ S[pos];
      }
   S[lower] ⇆ S[pos];
   return   pos;
}
```

Tìm kiếm nội suy



Nếu gọi k là giá trị cần tìm ($a_l \le k \le a_r$) thì vị trí x "tương ứng với k" được xác định bởi công thức sau:

$$\frac{x-l}{r-l}=\frac{k-a_l}{a_r-a_l} \Longrightarrow x=l+\left\lfloor\frac{(k-a_l)(r-l)}{a_r-a_l}\right\rfloor$$

Có được giá trị x, chúng ta hy vọng a_x sẽ là phần tử cần tìm:

- Nếu $a_x = k$: Dừng tìm kiếm.
- Nếu $a_x > k$: Giải thuật tiếp tục tìm kiếm trong đoạn l và $(r =)x - 1$.
- Nếu $a_x < k$: Đoạn tìm kiếm kế tiếp là $(l =)x + 1$ và r.