ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

BÁO CÁO BÀI TẬP

QUY HOẠCH ĐỘNG - PHẦN 2

Giảng viên: ThS. Nguyễn Thanh Sơn

Lớp: CS112.N21.KHTN

Thành viên: Hà Văn Hoàng - 21520033

Võ Thị Phương Anh - 21522883

Ngày: 15/06/2023

Mục lục

1. Bài tập

1.1. Yêu cầu:

Chỉ ra đâu là optimal substructure (cấu trúc con tối ưu) và overlapping subproblems (các bài toán con gối nhau) trong 3 bài toán mà nhóm đăng trên wecode.

1.2. Lời giải:

a. Đối với bài Repost, nhận thấy đây là bài toán làm việc trên đồ thị DAG, nên ta đặt len_v là độ dài chuỗi chia sẻ dài nhất và kết thúc tại tên v. Khi đó ta có công thức: $len_v = max(len_u) + 1$, với u có thể chia sẻ đến v. Nhận thấy đây là bài toán có các bài toán con gối nhau do nhận thấy khi xét theo góc nhìn đệ quy, lúc tìm len_v ta phải tính len_u, nếu u chia sẻ cho nhiều v, ta phải tính len_u nhiều lần. Và ta thấy có cấu trúc con tối ưu do khi giải được bài toán tối ưu tìm len_u, ta có thể lấy kết quả này để cập nhật lên bài toán tối ưu lớn hơn là tìm len_v.

b. Đối với bài Thương uống Mixue, ta có thể đặt dp_mask là chi phí ít nhất để đổ các ly không thuộc mask vào các ly thuộc mask ($mask_i = 1$: cốc i đã được đổ sang cốc khác và bị loại bỏ sau đó (không thuộc mask); $mask_i = 0$: ngược lại. Công thức tổng quát như sau:

$$dp_{mask} = min(dp_{submask} + cost_{u,v})$$

Với $submask$ có số lượng bit 1 của $mask$ hơn số lượng bit 1 của $submask$ đúng 1, u không thuộc mask và v thuộc $mask$, $submask$ or $(1 << u) = mask$. Dễ thấy đây cũng là bài toán có các bài toán gối nhau do dp_{mask} sẽ được tính dựa vào các $submask$ tương ứng. Mà mỗi $submask$ lại có thể có nhiều $mask$ thoả mãn, nên khi tính các dp_{mask} đó, thì bài toán tính $dp_{submask}$ sẽ được gọi nhiều lần, nên sẽ phải tính nhiều lần. Xét thấy đây cũng là bài toán có cấu trúc con tối ưu do từ bài toán con tối ưu $dp_{submask}$ có thể cập nhật kết quả lên bài toán con

tối ưu lớn hơn dp_{mask}.

c. Đối với bài Bài toán thang máy, ta có thể đặt $dp_{mask}.R$ là số chuyến ít nhất và $dp_{mask}.W$ là khối lượng ít nhất trong số chuyến cuối cùng để đưa tất cả mọi người thuộc $mask$ lên tầng cao nhất trong trường hợp các R đều ít nhất ($mask_i = 1$: người i đã lên tầng cao nhất; $mask_i = 0$: ngược lại).

Có hai trường hợp xảy ra khi thêm người i thuộc $mask$ vào thang máy (đặt $submask = mask$ xor $(1 << i)$, i không thuộc $submask$:

- Nếu $dp_{submask}.W + weight(i) > x$: Ta phải lập chuyến mới để đưa i lên, khi đó ta sẽ cập nhật kết quả bộ $\{dp_{submask}.R + 1, weight(i)\}$ vào dp_{mask}.

- Nếu $dp_{submask}.W + weight(i) <= x$: Ta đưa i vào chung với chuyến đi cuối cùng đó, khi đó ta sẽ cập nhật kết quả bộ $\{dp_{submask}.R, dp_{submask}.W + weight(i)\}$ vào dp_{mask}.

Dễ thấy đây cũng là bài toán có các bài toán gối nhau do dp_{mask} sẽ được tính dựa vào các $submask$ tương ứng. Mà mỗi $submask$ lại có thể có nhiều $mask$ thoả mãn, nên khi tính các dp_{mask} đó, thì bài toán tính $dp_{submask}$ sẽ được gọi nhiều lần, nên sẽ phải tính nhiều lần. Xét thấy đây cũng là bài toán có cấu trúc con tối ưu do từ bài toán con tối ưu $dp_{submask}$ có thể cập nhật kết quả lên bài toán con tối ưu lớn hơn dp_{mask}.

Để hiểu rõ hơn cách quy hoạch động 3 bài trên, có thể xem code được submit bởi "Hà Văn Hoàng" trên wecode.