CS 170 3.2 Lecture 12

Knapsack with Repetition

We can pick the same item more than once.

Subproblems: $K(w)$ = max value achievable with capacity $\leq w$

Boundaries: $K(0) = 0$

Recurrence: $K(w) = \max_{i=1\cdots n} \{ K(w - w_i) + v_i \mid w_i \leq w \}$

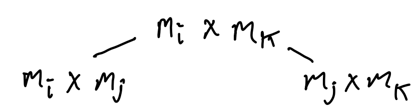
Efficiency: solve in increasing order of weight w: $K(0), K(1), \ldots, K(W)$
each computation takes $O(n)$ time
total time is $O(n \cdot W)$, same as without repetition

This is weakly polynomial algorithm — polynomial would be poly$(n \log W)$
Problem is NP-complete, so we don't expect a polynomial runtime.

Chain Matrix Multiplication

We want to multiply $A_1, A_2, \ldots, A_n$ as cheaply as possible.
Multiplying x-by-y and y-by-z requires xyz operations.
Let A = 50-by-1, B = 1-by-50, and C = 50-by-1. Then, $(AB)C$ = 5000 ops, but $A(BC)$ = 100 operations. Order matters! Let's find an optimal association order.

Input: $m_0, m_1, \ldots, m_n$ matrix dimensions
Output: parenthetization of / tree on list



Subproblems: $C(i, k)$ = optimal cost for subsequence $m_i, \ldots, m_K$ to multiply $A_{i+1} \ldots A_K$

Boundaries: $C(0,1) = 0$, $C(1,2) = 0$, $\ldots$ $C(n-1, n) = 0$

Recurrence: $C(i, k) = \min_{i<j<k} \{ C(i,j) + C(j,k) + m_i m_j m_K \}$

Cost: $O(n^2)$ subproblems, and each computation is $O(n) \longrightarrow O(n^3)$ $[C(0,n)]$

All-Pairs Shortest Paths

Find $\{\text{dist}(u,v)\}_{u,v \in V}$. Running Bellman-Ford from each vertex $= O(|V|^2 |E|)$
Floyd-Warshall algorithm $\longrightarrow O(|V|^3)$ $\quad$ ($|E| \geq |V| - 1$ if G is connected)

Subproblems: $d(u, v, i)$ = shortest path (u,v) using nodes only in $\{1, \ldots, i\}$

Boundaries: $d(u,v,0) = \begin{cases} (u,v) \in E : \ell(u,v) \\ (u,v) \notin E : \infty \end{cases}$

Recurrence: $d(u,v,i) = \min \{ d(u,v,i-1),\ d(u,i,i-1) + d(i,v,i-1) \}$

Cost: init boundaries: $O(|V|^2)$
table has $|V|^3$ entries, and computing each is $O(1)$
} total runtime is $O(|V|^3)$

TRAVELING SALESPERSON

We want to find the shortest tour in G.

Naively, we start WLOG at vertex 1. There $\leq (n-1)!$ tours, and trying each costs $O(n)$. Try them all, for a runtime $O(n!) \approx O(n^n/e^n)$

We can achieve $O(2^n \cdot n^2)$ via DP! This problem is NP-complete.

Subproblems: $C(S, j)$ = shortest path from 1 to j visiting each $v \in S$ once (assume $1 \in S$ but $j \notin S$)

using this, we can infer the tour: $\min_j \{ C(\{1, \dots, n\}, j) + \ell(j, 1) \}$

Boundaries: $C(\{1\}, 1) = 0$, $C(S, 1) = \infty$ if $|S| > 1$

Recurrence: $C(S, j) = \min_{i \in S, i \neq j} \{ C(S \setminus \{i\}, i) + \ell(i, j) \}$

Cost: compute in order of increasing $|S|$

table has $O(2^n \cdot n)$ entries, and each costs $O(n)$ to compute $\longrightarrow O(2^n \cdot n^2)$