CSL 356: Analysis and Design of Algorithms

Ragesh Jaiswal

CSE, IIT Delhi

Dynamic Programming: Examples

All pairs shortest paths

Dynamic Programming: Examples

Matrix Chain Multiplication

Dynamic Programming: Examples

- Problem(matrix chain multiplication): You are given a sequence of n matrices $M_1, \ldots, M_n$ of size $(m_1 \times m_2), (m_2 \times m_3), \ldots, (m_n \times m_{n+1})$. Determine in what order these matrices should be multiplied (using naïve method) so as to reduce the total running time.
- Example: Consider four matrices of size
  - M_1: 50×20
  - M_2: 20×1
  - M_3: 1×10
  - M_4: 10×100

- $M_1 \times M_2 \times M_3 \times M_4 = M_1 \times ((M_2 \times M_3) \times M_4)$
  - Time:
- $M_1 \times M_2 \times M_3 \times M_4 = (M_1 \times (M_2 \times M_3)) \times M_4$
  - Time:
- $M_1 \times M_2 \times M_3 \times M_4 = (M_1 \times M_2) \times (M_3 \times M_4)$
  - Time:

Dynamic Programming: Examples

- Problem(matrix chain multiplication): You are given a sequence of n matrices $M_1, \ldots, M_n$ of size $(m_1 \times m_2), (m_2 \times m_3), \ldots, (m_n \times m_{n+1})$. Determine in what order these matrices should be multiplied (using naïve method) so as to reduce the total running time.
- Example: Consider four matrices of size
  - M_1: 50×20
  - M_2: 20×1
  - M_3: 1×10
  - M_4: 10×100

- $M_1 \times M_2 \times M_3 \times M_4 = M_1 \times ((M_2 \times M_3) \times M_4)$
  - Time: $20 \cdot 10 + 20 \cdot 10 \cdot 100 + 50 \cdot 20 \cdot 100$
- $M_1 \times M_2 \times M_3 \times M_4 = (M_1 \times (M_2 \times M_3)) \times M_4$
  - Time: $20 \cdot 10 + 50 \cdot 20 \cdot 10 + 50 \cdot 10 \cdot 100$
- $M_1 \times M_2 \times M_3 \times M_4 = (M_1 \times M_2) \times (M_3 \times M_4)$
  - Time: $50 \cdot 20 + 10 \cdot 100 + 50 \cdot 100$

Dynamic Programming: Examples

- $C(i,j)$: Minimum cost of multiplying matrices $M_i, \ldots, M_j$.
- $C(i,i) = 0$
- $C(i,j)$?

Dynamic Programming: Examples

- $C(i,j)$: Minimum cost of multiplying matrices $M_i, \dots, M_j$.
- $C(i,i) = 0$
- $C(i,j) = \min_{i \le k < j} \left(C(i,k) + C(k+1,j) + m_i \cdot m_{k+1} \cdot m_{j+1} \right)$

Matrix-Cost($M_1, \dots, M_n$)
- for $i = 1$ to n
  - $C[i,i] = 0$
- for $s = 1$ to $n - 1$
  - for $i = 1$ to $n - s$
    - $j = i + s$
    - $C[i,j] = \min_{i \le k < j} \left(C[i,k] + C[k+1,j] + m_i \cdot m_{k+1} \cdot m_{j+1} \right)$
- return($C[1,n]$)

- Running time:

Dynamic Programming: Examples

- $C(i,j)$: Minimum cost of multiplying matrices $M_i, \ldots, M_j$.
- $C(i,i) = 0$
- $C(i,j) = \min_{i \le k < j} \big(C(i,k) + C(k+1,j) + m_i \cdot m_{k+1} \cdot m_{j+1} \big)$

Matrix-Cost($M_1, \ldots, M_n$)
- for $i = 1$ to n
    - $C[i,i] = 0$
- for $s = 1$ to $n - 1$
    - for $i = 1$ to $n - s$
        - $j = i + s$
        - $C[i,j] = \min_{i \le k < j} \big(C[i,k] + C[k+1,j] + m_i \cdot m_{k+1} \cdot m_{j+1} \big)$
- return($C[1,n]$)

- Running time: $O(n^3)$.

Dynamic Programming: Examples

- Problem (all pairs shortest path): Let $G = (V, E)$ be a weighted graph that has no negative cycles. Give an algorithm to find the shortest path between all pairs of vertices.

- Suppose the edge weights are positive.
  - Use Dijkstra's algorithm for all source vertices to find all pairs shortest paths.
    - Running time:

Dynamic Programming: Examples

- Problem (all pairs shortest path): Let $G = (V, E)$ be a weighted graph that has no negative cycles. Give an algorithm to find the shortest path between all pairs of vertices.

- Suppose the edge weights are positive.
  - Use Dijkstra's algorithm for all source vertices to find all pairs shortest paths.
    - Running time: $O(mn \log n)$
- Edge weights could be negative.
  - There is an algorithm called *Bellman-Ford* that computes single source shortest paths in time $O(nm)$. We will get an $O(n^2m)$ algorithm using this.

Dynamic Programming: Examples

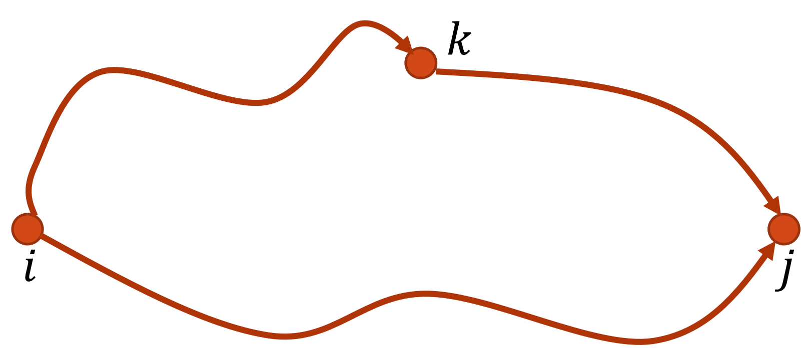
- Dynamic programming solution: *Floyd-Warshall*
  - Label the vertices in the graph $\{1,2,\ldots,n\}$
  - $L(i,j,k)$: The shortest path length between vertices i and j such that all intermediate vertices have labels $\{1,\ldots,k\}$.

Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - Label the vertices in the graph $\{1,2,\ldots,n\}$
  - $L(i,j,k)$: The shortest path length between vertices i and j such that all intermediate vertices have labels $\{1,\ldots,k\}$.
  - $L(i,j,0) = e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?

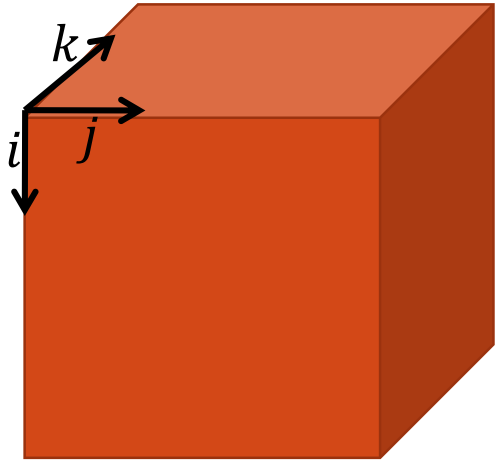
Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - Label the vertices in the graph $\{1,2,\ldots,n\}$
  - $L(i,j,k)$: The shortest path length between vertices i and j such that all intermediate vertices have labels $\{1,\ldots,k\}$.
  - $L(i,j,0) = e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?
    - $L(i,j,k) = min(L(i,j,k-1), L(i,k,k-1) + L(k,j,k-1))$



Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i,j,0) = e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?
    - $L(i,j,k) = min(L(i,j,k-1), L(i,k,k-1) + L(k,j,k-1))$
  - How do we fill the 3-D matrix **L**?



Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i,j,0) = e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?
    - $L(i,j,k) = min(L(i,j,k-1), L(i,k,k-1) + L(k,j,k-1))$
  - How do we fill the 3-D matrix L?
    - Compute $L(i,j,k)$ before computing $L(i,j,k+1)$.
  - How do we compute the shortest paths?

Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i,j,0) = e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?
    - $L(i,j,k) = min(L(i,j,k-1), L(i,k,k-1) + L(k,j,k-1))$
  - How do we fill the 3-D matrix L?
    - Compute $L(i,j,k)$ before computing $L(i,j,k+1)$.
  - How do we compute the shortest paths?
    - Can we keep storing the predecessor of j in the shortest path with intermediate vertices $\{1, \ldots, k\}$?

Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i, j, 0) = e(i, j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i, j, k)$ for $k > 0$?
    - $L(i, j, k) = min(L(i, j, k-1), L(i, k, k-1) + L(k, j, k-1))$
  - How do we fill the 3-D matrix L?
    - Compute $L(i, j, k)$ before computing $L(i, j, k+1)$.
  - How do we compute the shortest paths?
    - Can we keep storing the predecessor of j in the shortest path with intermediate vertices $\{1, \ldots, k\}$?
    - $P(i, j, k) =$ either $P(k, j, k-1)$ or $P(i, j, k-1)$

Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i,j,0) \ = \ e(i,j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i,j,k)$ for $k > 0$?
    - $L(i,j,k) \ = min(L(i,j,k-1), L(i,k,k-1) + L(k,j,k-1))$
  - How do we fill the 3-D matrix L?
    - Compute $L(i,j,k)$ before computing $L(i,j,k+1)$.
  - How do we compute the shortest paths?
    - Can we keep storing the predecessor of j in the shortest path with intermediate vertices $\{1, \ldots, k\}$?
    - $P(i,j,k) =$ either $P(k,j,k-1)$ or $P(i,j,k-1)$
  - What is the running time?

Dynamic Programming: Examples

- Dynamic programming solution: *Floyd-Warshall*
  - $L(i, j, 0) = e(i, j)$ if there is an edge between i and j else ∞ (a very large number).
  - What is the value of $L(i, j, k)$ for $k > 0$?
    - $L(i, j, k) = min(L(i, j, k-1), L(i, k, k-1) + L(k, j, k-1))$
  - How do we fill the 3-D matrix L?
    - Compute $L(i, j, k)$ before computing $L(i, j, k+1)$.
  - How do we compute the shortest paths?
    - Can we keep storing the predecessor of j in the shortest path with intermediate vertices $\{1, \ldots, k\}$?
    - $P(i, j, k) =$ either $P(k, j, k-1)$ or $P(i, j, k-1)$
  - What is the running time?
    - $O(n^3)$

End