CSE 565

LECTURES 15
Dynamic Programming
- RNA Secondary Structure
- Shortest Paths: Bellman-Ford

Sofya Raskhodnikova

10/17/2016
Review

• Weighted independent set on the chain
  – Input: a chain graph of length $n$ with values $v_1, \ldots, v_n$
  – Goal: find a heaviest independent set

Let OPT(j) = ???

Write down a recursive formula for OPT
  – How many different subproblems?
Exercise

- Do the same with a $2 \times n$ grid graph
Exercise

• Do the same with a 2 x n grid graph

• Three types of subproblems:
  – grid(i)
  – gridTop(i)
  – gridBottom(i)
Exercise

- grid(i): maximum independent set in the subgraph consisting of only the first i pairs of nodes

- gridTop(i): maximum independent set in the subgraph consisting of the first i pairs of nodes plus the top node of the (i+1)-st pair

- gridBottom(i): maximum independent set in the subgraph consisting of the first i pairs of nodes plus the bottom node of the (i+1)-st pair
RNA Secondary Structure
RNA Secondary Structure

**RNA.** String $B = b_1 b_2 \ldots b_n$ over alphabet \{ A, C, G, U \}.

**Secondary structure.** RNA is single-stranded so it tends to loop back and form base pairs with itself. This structure is essential for understanding behavior of molecule.

Ex: \texttt{GUCGAUUGAGCGAAUGUAACAACGUGGCUCACCGCGGAGA}

complementary base pairs: A-U, C-G
RNA Secondary Structure

Secondary structure. A set of pairs $S = \{ (b_i, b_j) \}$ that satisfy:

- [Watson-Crick.] $S$ is a matching and each pair in $S$ is a Watson-Crick complement: A-U, U-A, C-G, or G-C.
- [No sharp turns.] The ends of each pair are separated by at least 4 intervening bases. If $(b_i, b_j) \in S$, then $i < j - 4$.
- [Non-crossing.] If $(b_i, b_j)$ and $(b_k, b_l)$ are two pairs in $S$, then we cannot have $i < k < j < l$.

Free energy. Usual hypothesis is that an RNA molecule will form the secondary structure with the optimum total free energy.

Given an RNA molecule $B = b_1b_2\ldots b_n$, find a secondary structure $S$ that maximizes the number of base pairs.
RNA Secondary Structure: Examples

Examples.

[Diagrams of RNA secondary structures showing examples of base pairs, sharp turns, and crossings.]
RNA Secondary Structure: Subproblems

First attempt. \( \text{OPT}(j) = \) maximum number of base pairs in a secondary structure of the substring \( b_1b_2\ldots b_j \).

**Difficulty.** Results in two sub-problems.

- Finding secondary structure in: \( b_1b_2\ldots b_{t-1} \). \( \xrightarrow{\text{OPT}(t-1)} \)
- Finding secondary structure in: \( b_{t+1}b_{t+2}\ldots b_{n-1} \). \( \xrightarrow{\text{need more sub-problems}} \)
Dynamic Programming Over Intervals

Notation. \( \text{OPT}(i, j) = \text{maximum number of base pairs in a secondary structure of the substring } b_i b_{i+1} \ldots b_j. \)

- **Case 1.** \( i \geq j - 4. \)
  - \( \text{OPT}(i, j) = 0 \) by no-sharp turns condition.

- **Case 2.** Base \( b_j \) is not involved in a pair.
  - \( \text{OPT}(i, j) = \text{OPT}(i, j-1) \)

- **Case 3.** Base \( b_j \) pairs with \( b_t \) for some \( i \leq t < j - 4. \)
  - non-crossing constraint decouples resulting sub-problems
  - \( \text{OPT}(i, j) = 1 + \max_t \{ \text{OPT}(i, t-1) + \text{OPT}(t+1, j-1) \} \)
  - take max over \( t \) such that \( i \leq t < j-4 \) and \( b_t \) and \( b_j \) are Watson-Crick complements
Q. In what order should we solve the subproblems?

A. Do shortest intervals first.

```c
RNA(b_1, \ldots, b_n) \{
    for k = 5, 6, \ldots, n-1
        for i = 1, 2, \ldots, n-k
            j = i + k
            Compute M[i, j]
        return M[1, n]
    \}
```

Running time. \(O(n^3)\).
Dynamic Programming Summary

Recipe.
- Decide which subproblems to use (define OPT(??)). If it fails, try again.
- Recursively define value of optimal solution.
- Compute value of optimal solution (bottom up or via memoization).
- Construct optimal solution from computed information.

Dynamic programming techniques.
- Binary choice: weighted interval scheduling.
- Multi-way choice: segmented least squares.
- Adding a new variable: knapsack.
- Dynamic programming over intervals: RNA secondary structure.
- Viterbi algorithm for Hidden Markov Models also uses DP to optimize a maximum likelihood tradeoff between parsimony and accuracy
- Parsing algorithm for context-free grammars has similar structure

Top-down vs. bottom-up: different people have different intuition.
Bellman-Ford: Shortest paths via dynamic programming

For consistency with book: shortest paths from all vertices to a destination $t$
Single-source Shortest Path Problem

• **Input:**
  - Directed graph \( G = (V, E) \).
  - Source node \( s \), destination node \( t \).
  - for each edge \( e \), length \( \ell(e) = \text{length of } e \).
  - length path = sum of edge lengths

• **Find:** shortest directed path from \( s \) to \( t \).

Length of path \((s,2,3,5,t)\) is \( 9 + 23 + 2 + 16 = 50 \).
When is there a shortest path?

Under which conditions do shortest paths

– Always exist?
– Sometimes exist and sometimes not exist?
– Never exist?

• In a directed graph with nonnegative edge lengths?

• In a directed graph with negative edges lengths?
When is there a shortest path?

• Edge weights nonnegative:
  – shortest path always exists

• Negative weight edges:
  – shortest path may exist or not
  – If no negative cycles in $G$, then shortest path exists
  – If negative cycles in $G$, then no shortest path
Observation. If some path from s to t contains a negative cost cycle, there does not exist a shortest s-t path; otherwise, there exists one that is simple.
Shortest Paths: Dynamic Programming

Def. $OPT(i, v)$ = length of shortest $v$-$t$ path $P$ using at most $i$ edges.

- Case 1: $P$ uses at most $i-1$ edges.
  - $OPT(i, v) = OPT(i-1, v)$

- Case 2: $P$ uses exactly $i$ edges.
  - if $(v, w)$ is first edge, then $OPT$ uses $(v, w)$, and then selects best $w$-$t$ path using at most $i-1$ edges

Remark. By previous observation, if no negative cycles, then $OPT(n-1, v) = length$ of shortest $v$-$t$ path.

$$OPT(i, v) = \begin{cases} 
0 & \text{if } i = 0 \\
\min \left\{ OPT(i-1, v), \min_{(v, w) \in E} \{ OPT(i-1, w) + c_{vw} \} \right\} & \text{otherwise}
\end{cases}$$
Shortest Paths: Implementation

Shortest-Path(G, t) {
    foreach node v ∈ V
        M[0, v] ← ∞
    M[0, t] ← 0

    for i = 1 to n-1
        foreach node v ∈ V
            M[i, v] ← M[i-1, v]
        foreach edge (v, w) ∈ E
            M[i, v] ← min { M[i, v], M[i-1, w] + c_{vw} }
}

Analysis. Θ(mn) time, Θ(n^2) space.

Finding the shortest paths. Maintain a "successor" for each table entry.
Shortest Paths: Improvements

- Maintain one array $M[v] =$ length of shortest $v$-$t$ path found so far.
- No need to check edges of the form $(v, w)$ unless $M[w]$ changed in previous iteration.

**Theorem.** Throughout the algorithm,
- $M[v]$ is length of some $v$-$t$ path, and
- For every $i$: after $i$ rounds of updates, the value $M[v]$ is no larger than the length of shortest $v$-$t$ path using $\leq i$ edges.

**Space and time complexity.**
- Memory: $O(m + n)$.
- Running time: $O(mn)$ worst case, but substantially faster in practice.
Bellman-Ford: Efficient Implementation

Bellman-Ford-Shortest-Path(G, s, t) {
    foreach node v \in V {
        M[v] \leftarrow \infty
        successor[v] \leftarrow \emptyset
    }

    M[t] = 0
    for i = 1 to n-1 {
        foreach node w \in V {
            if (M[w] has been updated in previous iteration) {
                foreach node v such that (v, w) \in E {
                    if (M[v] > M[w] + c_{vw}) {
                        M[v] \leftarrow M[w] + c_{vw}
                        successor[v] \leftarrow w
                    }
                }
            }
        }
        If no M[w] value changed in iteration i, stop.
    }
}

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
The demonstration is for a slightly different version of the algorithm (see CLRS) that computes distances from the source node rather than distances to the destination node.
Example of Bellman-Ford

Initialization.
Example of Bellman-Ford

Order of edge relaxation.
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford
Example of Bellman-Ford

\[\begin{array}{cccccc}
A & B & C & D & E \\
0 & 4 & 5 & 4 & \infty \\
-1 & 7 & 3 & 3 & 1 \\
-1 & 1 & 1 & 2 & 2 \\
\infty & 2 & 6 & 5 & -3 \\
\infty & \infty & \infty & \infty & \infty \\
\end{array}\]
Example of Bellman-Ford
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford
Example of Bellman-Ford

End of pass 1.

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

A – 1 2 4
B – 1 1 2
C – 4 3 1
D – 2 6 2
E – 1 8 3

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford
Example of Bellman-Ford
Example of Bellman-Ford

\begin{center}
\begin{tikzpicture}
  \node [circle,fill=yellow,draw] (A) at (0,0) {$A$};
  \node [circle,fill=yellow,draw] (B) at (2,1) {$B$};
  \node [circle,fill=yellow,draw] (C) at (1,-1) {$C$};
  \node [circle,fill=yellow,draw] (D) at (3,-1) {$D$};
  \node [circle,fill=yellow,draw] (E) at (3,0) {$E$};

  \path
    (A) edge [->,red,thick] node [above] {$-1$} (B)
    (B) edge [->] node [left] {$-1$} (A)
    (B) edge [->] node [right] {$1$} (C)
    (B) edge [->] node [right] {$2$} (D)
    (B) edge [->] node [right] {$1$} (E)
    (A) edge [->] node [right] {$5$} (C)
    (A) edge [->] node [right] {$4$} (D)
    (A) edge [->] node [right] {$4$} (E)
    (C) edge [->] node [right] {$3$} (D)
    (C) edge [->] node [right] {$5$} (E)
    (D) edge [->] node [right] {$3$} (E);
\end{tikzpicture}
\end{center}

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

A – 1
B – 2
C – 3
D – 4
E – 5

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne
Example of Bellman-Ford

End of pass 2 (and 3 and 4).

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne