

# Algorithm Design and Analysis

CSE  
565

## LECTURE 22 Dynamic Programming

- RNA Secondary Structure
- Shortest Paths

Sofya Raskhodnikova

10/17/2007

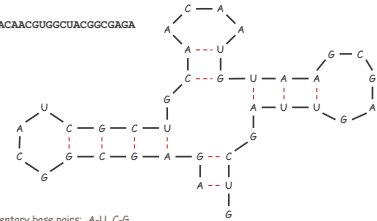
S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, K. Wayne

### RNA Secondary Structure Problem

RNA. String  $B = b_1b_2\dots b_n$  over alphabet  $\{A, C, G, U\}$ .

Secondary structure. RNA tends to loop back and form base pairs with itself. This structure is essential for understanding behavior of molecule.

Ex: GUCGAUUGAGCGAAUGUAACACGUGGCUACGGCGAGA



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  $i < k < j < l$ .

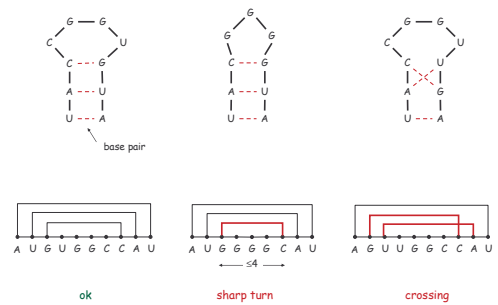
Free energy. Usual hypothesis is that an RNA molecule will form the secondary structure with the optimum total free energy.

approximate by number of base pairs

Goal. Given an RNA molecule  $B = b_1b_2\dots b_n$ , find a secondary structure  $S$  that maximizes the number of base pairs.

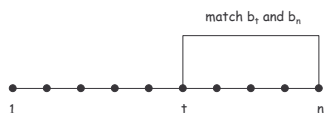
### RNA Secondary Structure: Examples

Examples.



### RNA Secondary Structure: Subproblems

First attempt.  $OPT(j) =$   
maximum number of base pairs in a secondary structure of the substring  $b_1b_2\dots b_j$ .



Difficulty. Results in two sub-problems.

- Finding secondary structure in:  $b_1b_2\dots b_{t-1}$ .  $\leftarrow OPT(t-1)$
- Finding secondary structure in:  $b_{t+1}b_{t+2}\dots b_{n-1}$ .  $\leftarrow$  need more sub-problems

### Dynamic Programming Over Intervals

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

- Base case. If  $i \geq j - 4$ .  
-  $OPT(i, j) = 0$  by no-sharp turns condition.
- Case 1. Base  $b_j$  is not involved in a pair.  
-  $OPT(i, j) = OPT(i, j-1)$
- Case 2. Base  $b_j$  pairs with  $b_t$  for some  $i \leq t < j - 4$ .  
- non-crossing constraint decouples resulting sub-problems  
-  $OPT(i, j) = 1 + \max_t \{ OPT(i, t-1) + OPT(t+1, j-1) \}$

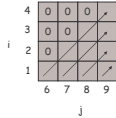
take max over  $t$  such that  $i \leq t < j - 4$  and  $b_t$  and  $b_j$  are Watson-Crick complements

### Bottom Up Dynamic Programming Over Intervals

Q. What order to solve the sub-problems?  
 A. Do shortest intervals first.

```

RNA(b1, ..., bn) { %Ignoring base cases
  for k = 5, 6, ..., n-1
    for i = 1, 2, ..., n-k
      j = i + k
      Compute M[i, j]
  return M[1, n]
}
    
```



Time:  $O(n^3)$ .  
 Space:  $O(n^2)$ .  
**Exercise:** Find the secondary structure that achieves the max value.

### Dynamic Programming Summary

- Recipe.
- Recursively define value of optimal solution.
  - Compute value of optimal solution.
  - Construct optimal solution from computed information.

- Dynamic programming techniques.
- Binary choice: weighted interval scheduling.
  - Multi-way choice: segmented least squares. [KT, section 6.3]
  - Adding a new variable: LCS, knapsack.
  - Dynamic programming over intervals: RNA secondary structure.

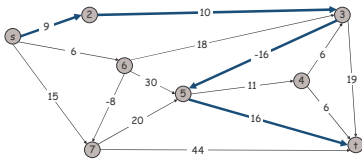
↙ parsing algorithm for context-free grammar has similar structure

Top-down (memoization) vs. bottom-up: different people have different intuitions.

### Shortest Paths

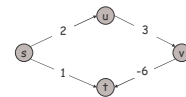
Shortest path problem. Given a directed graph  $G = (V, E)$ , with edge weights  $c_{uv}$ , find shortest path from node  $s$  to node  $t$ .  
 ↙ allow negative weights

Ex. Nodes represent agents in a financial setting and  $c_{vw}$  is cost of transaction in which we buy from agent  $v$  and sell immediately to  $w$ .



### Shortest Paths: Failed Attempts

Dijkstra. Can fail if there are negative edge costs.



Re-weighting. Adding a constant to every edge weight can fail.

