

# Alignment Space Saving Techniques

BCB 567, Fall 2007

# What is theoretical minimum space?

- Need to store string A:  $O(n)$
- Need to store string B:  $O(m)$
- Therefore the total amount of space needed to store the input is  $O(n+m)$
- We cannot do better than this, but can we match it.

# Finding the Score in $O(n+m)$ space

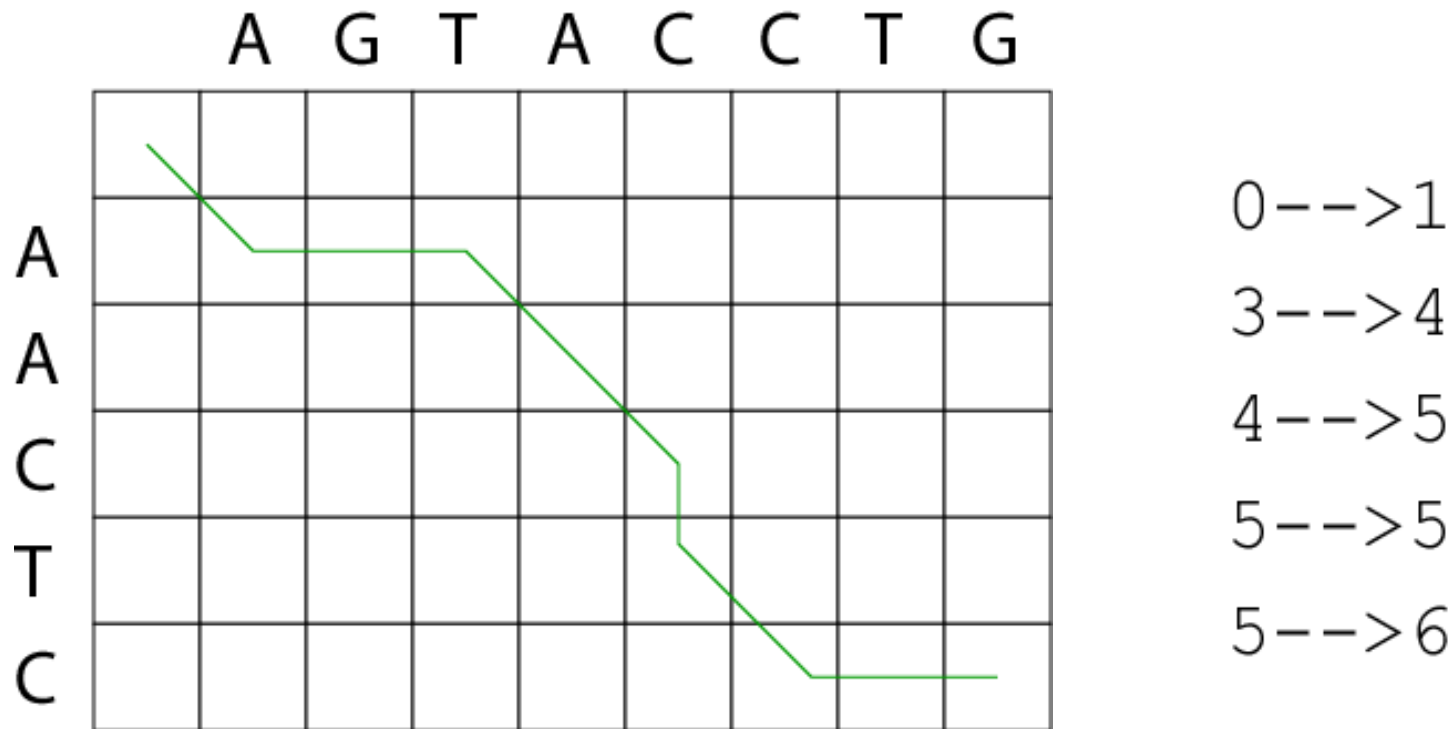
- How many rows of the table do we need to calculate the score?

```
for ( $i = 0; i \leq n; i++$ )  
     $row1[i] = i \times gamma$   
for ( $i = 0; i < n; i++$ )  
     $row2[0] = (i+1) \times gamma$ ;  
    calculate  $row2$  from  $row1$   
     $row1 = row2$   
 $score = row2[m]$ 
```

# Representing the Path in $O(n+m)$ space

- What is the maximum size of the path (in number of moves)
  - $O(n+m)$
- How can we encode the path in  $O(n+m)$  space?

# Intersection List Representation



A--ACTC--  
AGTAC-CTG

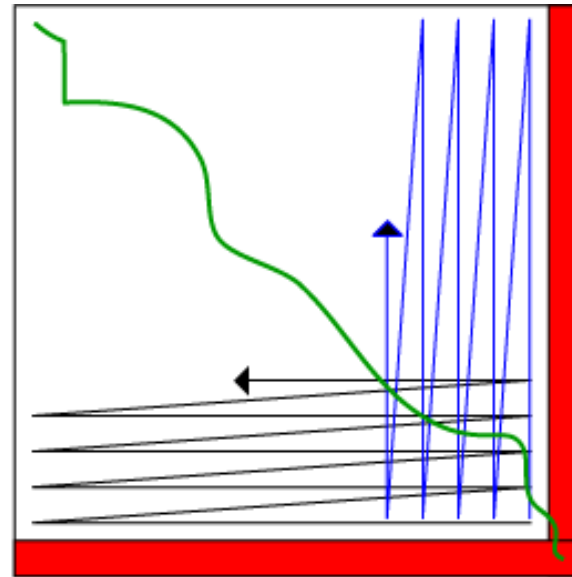
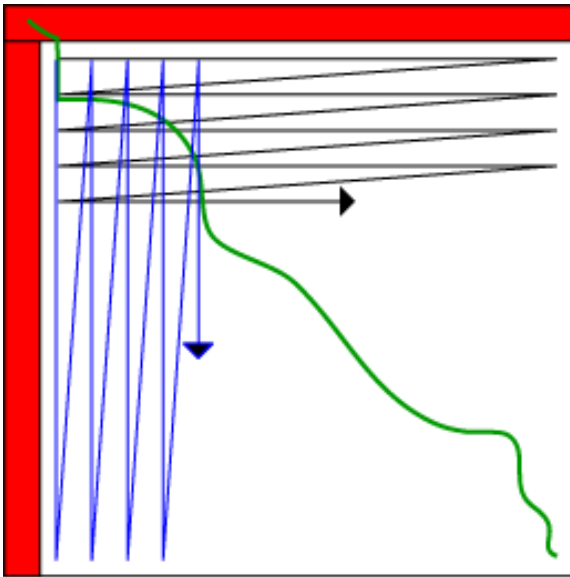
# Intersection list

- $n-1$  pairs of integers
  - $O(n+m)$  space
- Can construct the full alignment path from the intersection list
  - we know the start and end cells
  - can fill in horizontal moves

# Naïve Interval list construction in $O(n+m)$ space

- Build the interval list one interval at a time.
  - Can find the last interval by storing two rows of the table
  - Once the last interval has been found, reduce the size of the problem by one row (and  $k$  columns)
  - Continue until all intervals have been found.
- $O(n^3)$  time

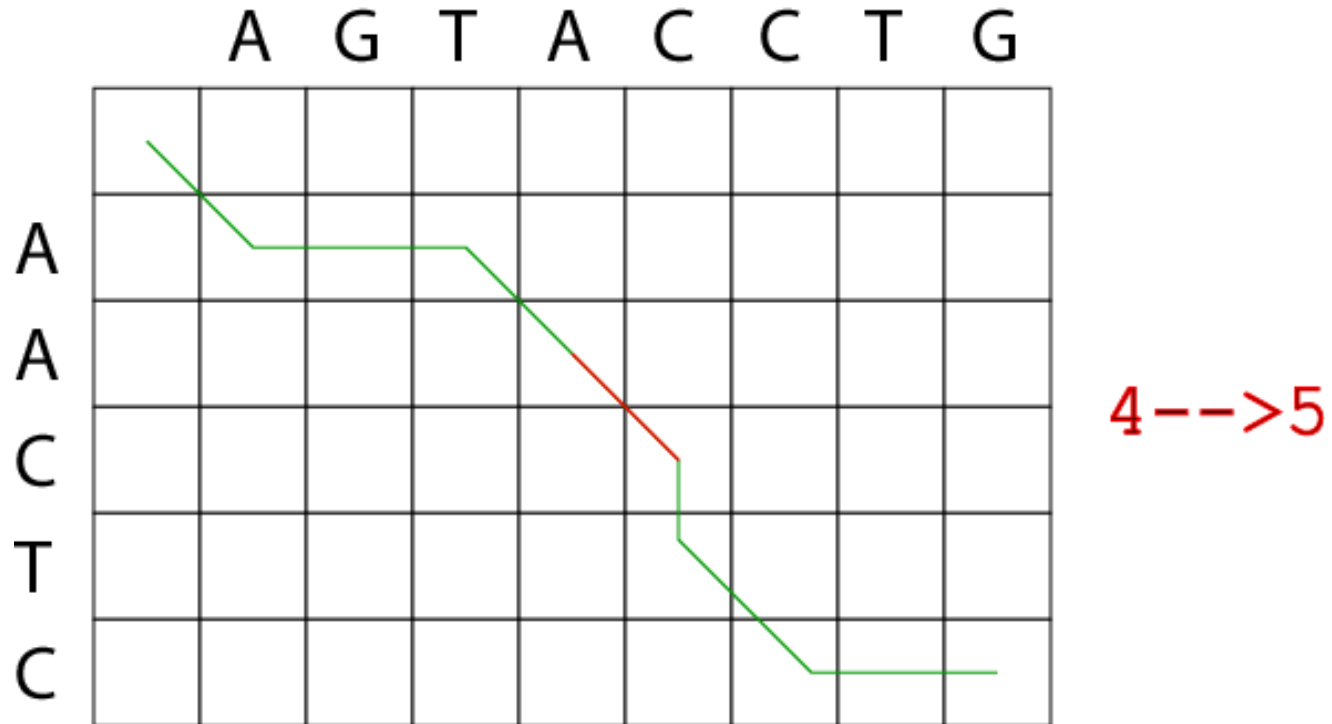
# Equivalent Solutions



$$S[i, j] = \max \left\{ \begin{array}{l} S[i-1, j-1] + \delta(i-1, j-1) \\ S[i-1, j] + \gamma \\ S[i, j-1] + \gamma \end{array} \right\}$$

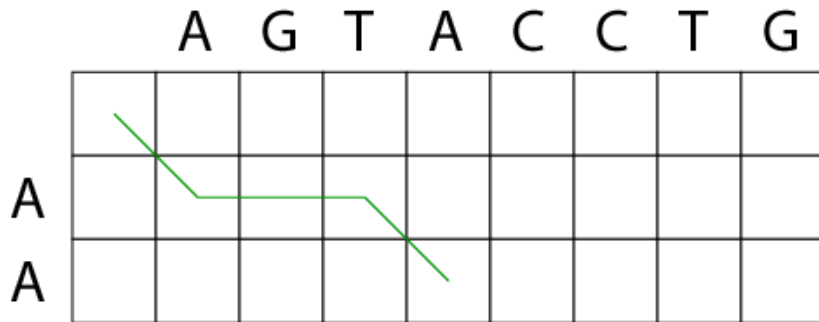
$$S[i, j] = \max \left\{ \begin{array}{l} S[i+1, j+1] + \delta(i, j) \\ S[i+1, j] + \gamma \\ S[i, j+1] + \gamma \end{array} \right\}$$

# Finding a Middle Intersection



A--ACTC--  
AGTAC-CTG

# Finding the Intersection with Table Expansion

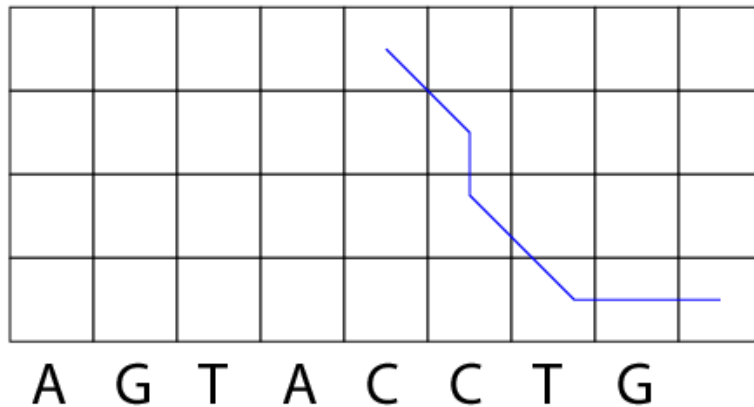


Find the maximum of

$$T1[2,i] + T2[0,i]$$

This is the score of the optimal alignment path.

A--A  
AGTA

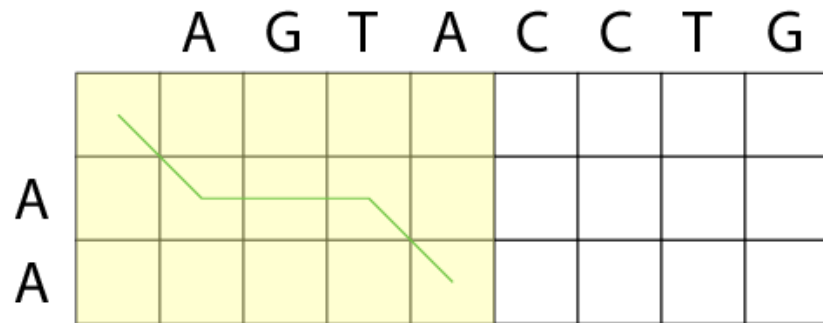


C  
T  
C

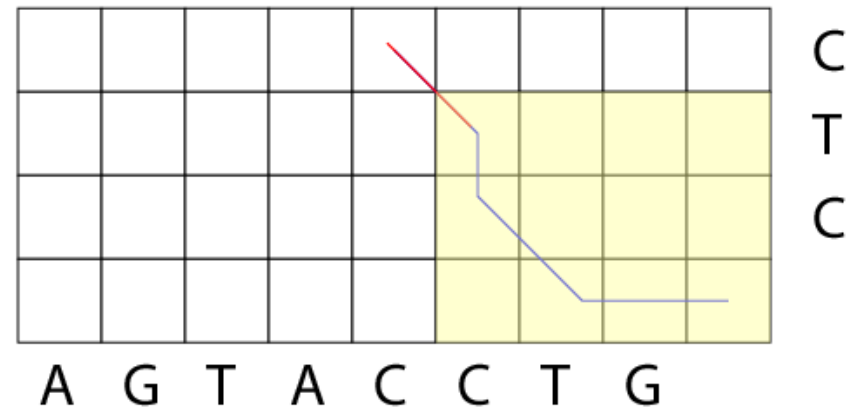
CTC--  
C-CTG

# The Intersection

- The first move of the trace-back in the bottom table is the intersection
- Found by storing only two rows of the table



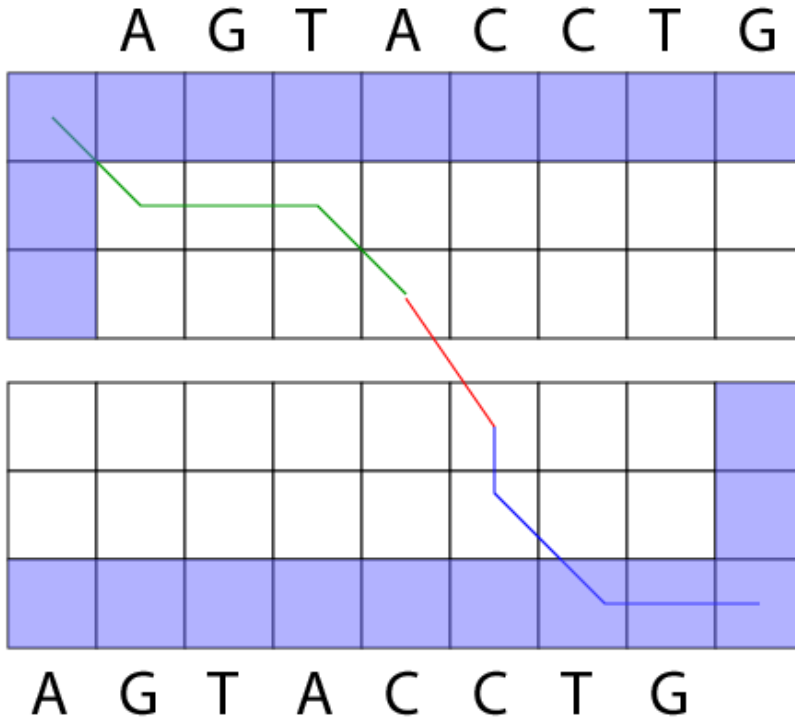
A--A  
AGTA



CTC--  
C-CTG

# Doing the recursion without table expansion

A--A  
AGTA

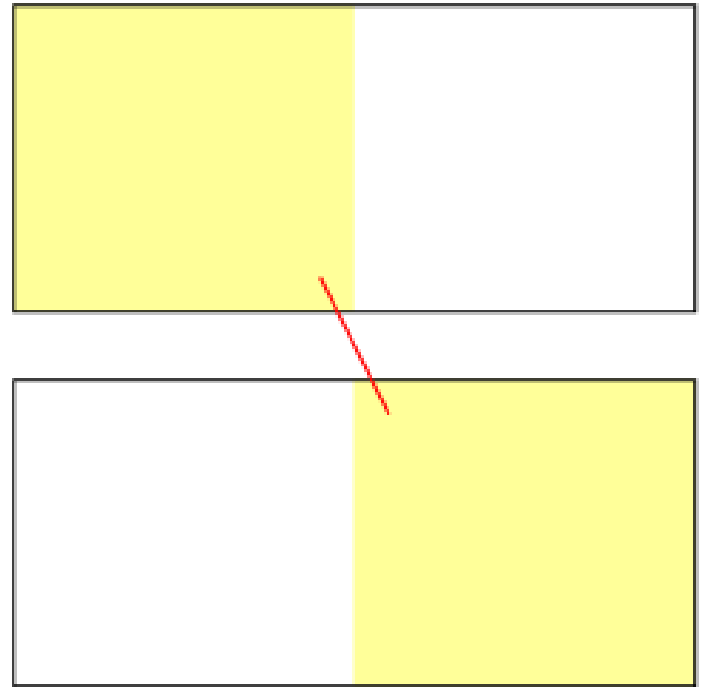
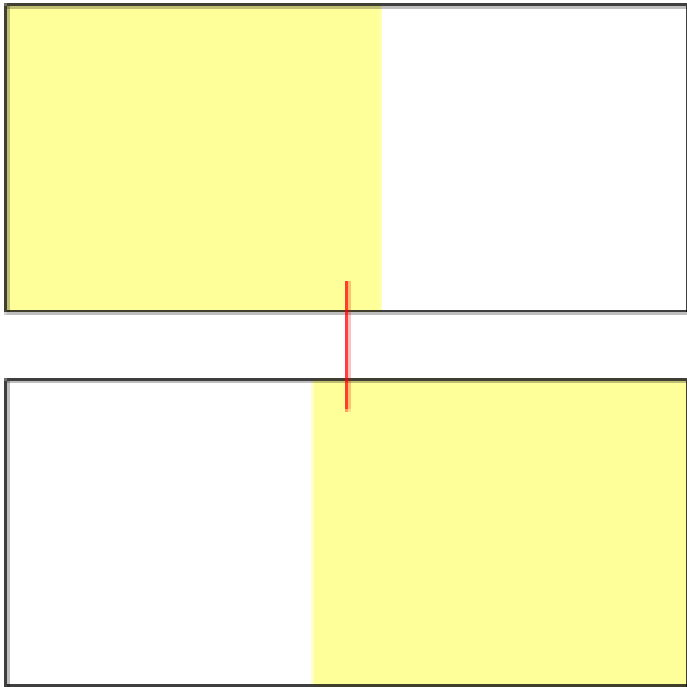


- The intersection between rows  $i$  and  $i+1$

TC--  
-CTG

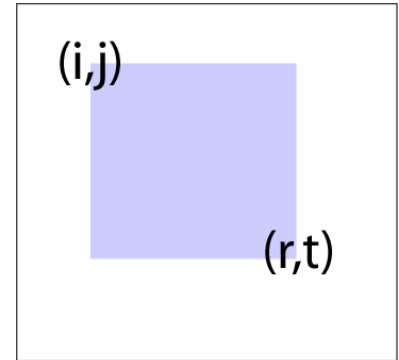
$$\underset{j=0}{\overset{m}{\text{Max}}} \left\{ \begin{array}{ll} S[i, j] + S[i+1, j] + \gamma & \text{intersection : } (j, j) \\ S[i, j] + S[i+1, j+1] + \delta(i, j) & \text{intersection : } (j-1, j) \end{array} \right\}$$

# Defining the recursive problems

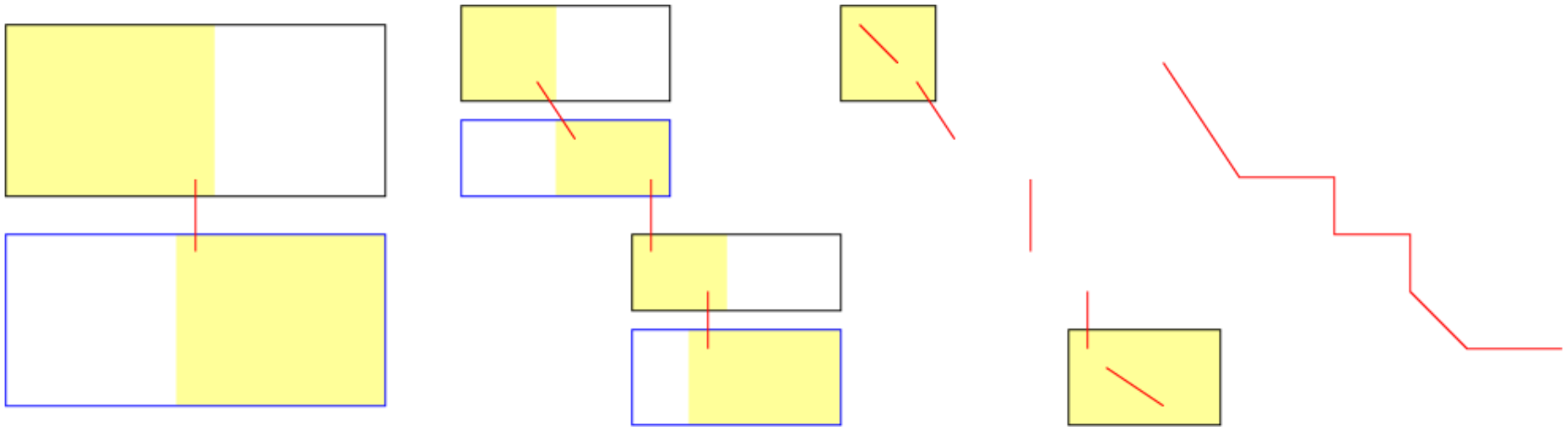


# Formal Definition of Solution

- Notation:  $S[i\dots r, j\dots t]$ 
  - $S[0\dots n, 0\dots m]$  entire table
- To solve  $S[i\dots r, j\dots t]$ 
  - Devide the problem into two halves:  
 $S[i\dots (i+r)/2, j\dots t]$  and  $S[(i+r)/2+1\dots r, j\dots t]$
  - Find the intersection between rows  $(i+r)/2$  and  $(i+r)/2 + 1$ , denoted as  $(f \rightarrow g)$
  - Recursively solve:
    - $S[i\dots (i+r)/2, j\dots f]$  and  $S[(i+r)/2+1\dots r, g\dots t]$



# Recursion



$$\sum 1 + \frac{1}{2} + \frac{1}{4} + \dots \approx 2$$