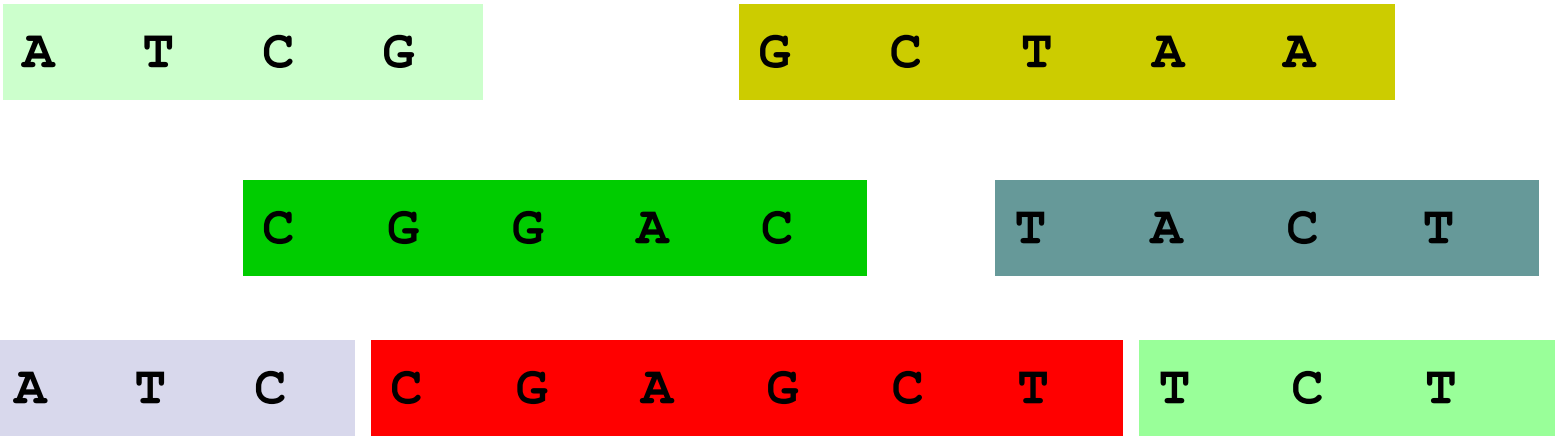
Shotgun Sequence Assembly

One example where beginning and ending gaps should be free is in "*shotgun sequence assembly*". In this problem, we have a large number of partially overlapping subsequences that come from many copies of one original but unknown DNA sequence. *The problem is to use comparisons of pairs of these subsequences to deduce the original DNA sequence.*

Two subsequences that are from different parts of the original sequence will have a low global alignment score as well as a low end-space free alignment score. Two overlapping subsequences will most likely still have a low global alignment score (since they are unlikely to have the same starting and ending position along the original sequence) but will have a *high end-space free alignment score!* The overlap will be detected and the subsequences will be pasted together to recover the original sequence using the alignments found.



Ends-Space Free Algorithm

1. Initialization Step. //initialize matrix

$F(0, 0) = 0$

$F(0, j) = 0$ //gaps are not penalized

$F(i, 0) = 0$ //gaps are not penalized

2. Main Iteration. //fill in table

for each $i = 1$ *to* m

for each $j = 1$ *to* n

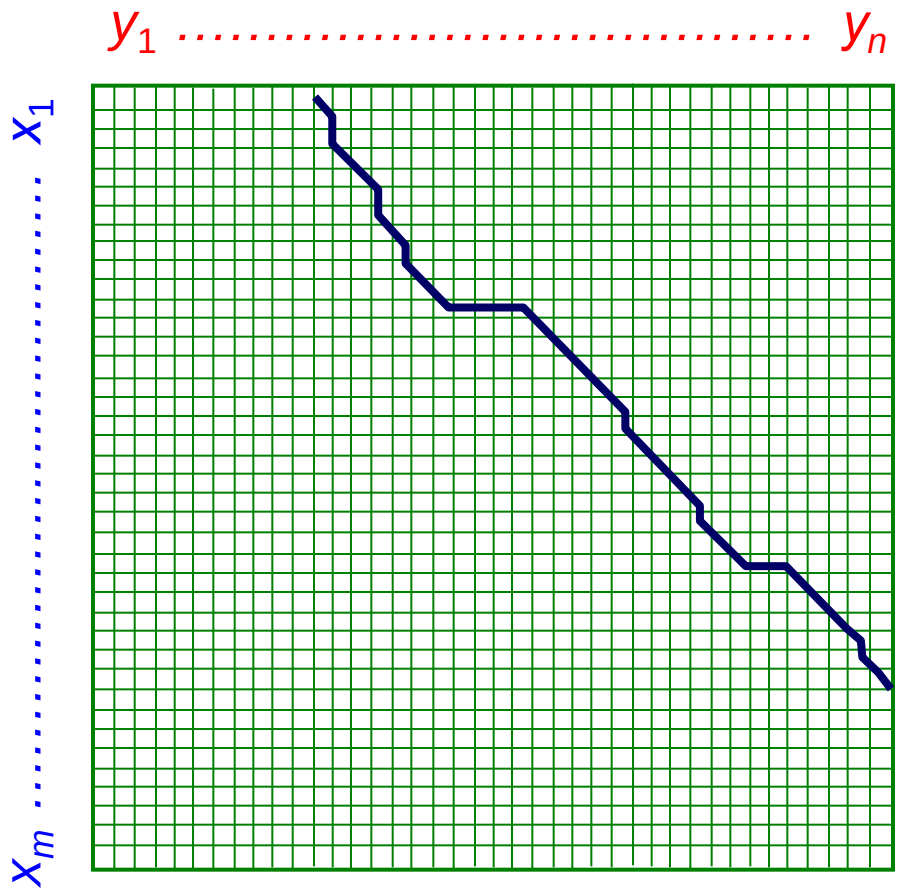
$$F(i, j) = \max \begin{cases} F(i-1, j-1) + m, \text{if } x_i = y_j & \text{(CASE 1)} \\ F(i-1, j-1) - s, \text{if } x_i \neq y_j & \text{(CASE 1)} \\ F(i-1, j) - d & \text{(CASE 2)} \\ F(i, j-1) - d & \text{(CASE 3)} \end{cases}$$

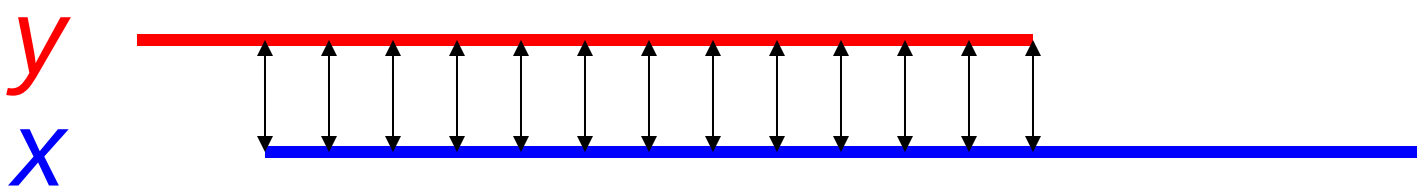
$$Ptr(i, j) = \begin{cases} \text{diagonal, if (CASE 1) holds} \\ \text{up, if (CASE 2) holds} \\ \text{left, if (CASE 3) holds} \end{cases}$$

3. Termination. //recover optimal score and alignment(s)

//return max{$F(i,n), F(m,j)$: $1 \le i \le m$, $1 \le j \le n$} as the optimal score S^*

//use $ptr(i,j)$ to reconstruct the optimal alignment(s)


y_1 ..... y_n
x_1
x_m


y
x

Example

$F(i,j)$

i	j = 0		1 A	2 C	3 T	4 G	5 T	6 T	7 A
0		0	0	0	0	0	0	0	0
1	G	0	-1	-1	-1	2	1	0	-1
2	T	0	-1	-2	1	1	4	3	2
3	T	0	-1	-2	0	0	3	6	5
4	A	0	2	1	0	-1	2	5	8
5	C	0	1	4	3	2	1	4	7
6	T	0	0	3	6	5	4	3	6
7	G	0	-1	2	5	8	7	6	5
8	T	0	-1	1	4	7	10	9	8

y = **ACTGTTA**
x = **GTTACTGT**

m = 2
s = d = 1

y = **---ACTGTTA**
x = **GTTACTGT--**