String Alignment 1
Dynamic Programming

How similar are these strings?

TGGCAGTCCCAGAAGGACTCTCCTC

TGCACAGTCCAGACACGGTCTCGTT

How similar are these strings?

TGGCAGTCCCAGAAGGACTCTCCTC

TGCACAGTCCAGACACGGTCTCGTT

- Phylogenetic ancestry
- Mutations due to disease
- Understanding gene function

String Alignment

G	C	A	G	T	C	C	G	A	C
G	C	G	T	C	T	G	A	C	T

#Matches: 2
#Mismatches: 8

String Alignment

G	C	A	G	T	C	C	G	A	C	
G	C		G	T	C	T	G	A	C	T

#Matches: 8

#Mismatches: 3

String Alignment

G	C	A	G	T	C	C	G	A	C	
G	C		G	T	C	T	G	A	C	T

A: Deleted C->T T: Inserted

Mutations: Errors in DNA replication? Disease?

String Similarity

G	C	A	G	T	C	C	G	A	C	
G	C		G	T	C	T	G	A	C	T

Minimum #Mismatches among all alignments

String Similarity

G	C	A	G	T	C	C	G	A	C	
G	C		G	T	C	T	G	A	C	T
M	M	D	M	M	M	R	M	M	M	I

Edit Distance = Minimum #
Insertions + Deletions + Replacements

Edit Distance

```
      GCAGTCCGAC
->    GCGTCCGAC
->    GCGTCTGAC
->    GCGTCTGACT
```

Minimum # Insertions + Deletions + Replacements

Edit Distance

Problem:

Input: Two strings S1 and S2

Output: Edit Distance

and

Optimal Edit Transcript/Alignment

Review:

ED(CLEARS, READS) =

ED(ACCTGCAA, CTGCAAG) =

ED(ϵ,GACCT)=

Dynamic Programming

Step 1: Identify subproblems to be solved.

Step 2: Develop a recurrence relation for each subproblem.

Step 3: Solve subproblems in bottom-up/tabular computation.

Step 4: Traceback to get solution.

Step 1: Subproblems

D(i,j)=Edit Distance between S1[1,…,i] and S2[1,…,j]

S1:TACCGCA S2: ACCGTAC

D(4,3)=ED(TACC,ACC)=1

D(2,2)=ED(TA,AC)=2

D(7,7)=ED(TACCGCA, ACCGTAC)

Step 2: Recurrence

S1:TACCGCA S2: ACCGTAC

Base Cases:

$D(0,j)=j$

$D(i,0)=i$

Step 2: Recurrence

S1:TACCGCA S2: ACCGTAC

ED(TAC, ACC)=ED(TA,AC)

ED(TACCGC, ACCGT)=1+Min{
    ED(TACCGC, ACCG),
    ED(TACCG,ACCGT),
    ED(TACCG,ACCG)}

Step 2: Recurrence

ED(TAC, ACC)=ED(AC,TA)

If S1[i]=S2[j], then D(i,j)=D(i-1,j-1)

Step 2: Recurrence

ED(TACCGC, ACCGT)=1+Min{
ED(TACCGC, ACCG),
ED(TACCG,ACCGT),
ED(TACCG,ACCG)}

If $S_1[i] \neq S_2[j]$, then:

$$D(i,j)=1+\mathrm{Min}\{D(i,j-1), D(i-1,j), D(i-1,j-1)\}$$

Step 2: Recurrence

If $S1[i]=S2[j]$, then

$$D(i,j)=D(i-1,j-1)$$

Else

$$D(i,j)=1+Min\{D(i,j-1), D(i-1,j),D(i-1,j-1)\}$$

Step 3: Tabular Computation

	D(i-1,j-1)	D(i-1,j)		
	D(i,j-1)	D(i,j)		

Step 3:
Tabular
Computation

	ϵ	A	C	C	G	T	A	C
ϵ	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
T	🟦	🟧	🟧	🟧	🟧	🟧	🟧	🟧
A	🟦	🟧	🟩	🟩	🟩	🟩	🟩	🟩
C	🟦	🟧	🟩					
C	🟦	🟧	🟩					
G	🟦	🟧	🟩					
C	🟦	🟧	🟩					
A	🟦	🟧	🟩					

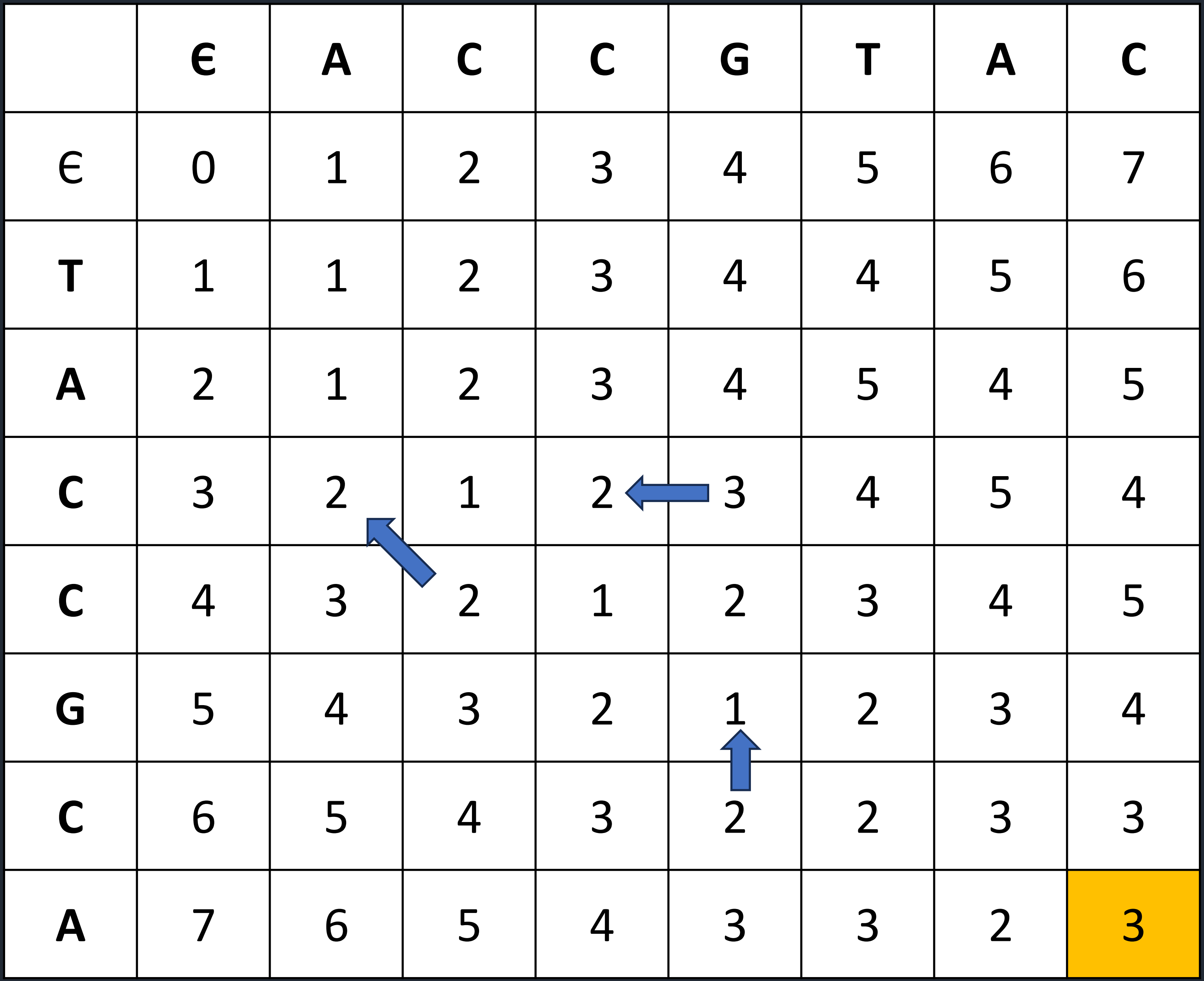
Step 3:
Tabular
Computation

	Є	A	C	C	G	T	A	C
Є	0	1	2	3	4	5	6	7
T	1	1	2	3	4	4	5	6
A	2	1						
C	3	2						
C	4	3						
G	5	4						
C	6	5						
A	7	6						

Step 3: Tabular Computation

	Є	**A**	**C**	**C**	**G**	**T**	**A**	**C**
Є	0	1	2	3	4	5	6	7
T	1	1	2	3	4	4	5	6
A	2	1	2	3	4	5	4	5
C	3	2	1	2	3	4	5	4
C	4	3	2	1	2	3	4	5
G	5	4	3	2	1	2	3	4
C	6	5	4	3	2	2	3	3
A	7	6	5	4	3	3	2	3

Step 4:
Traceback

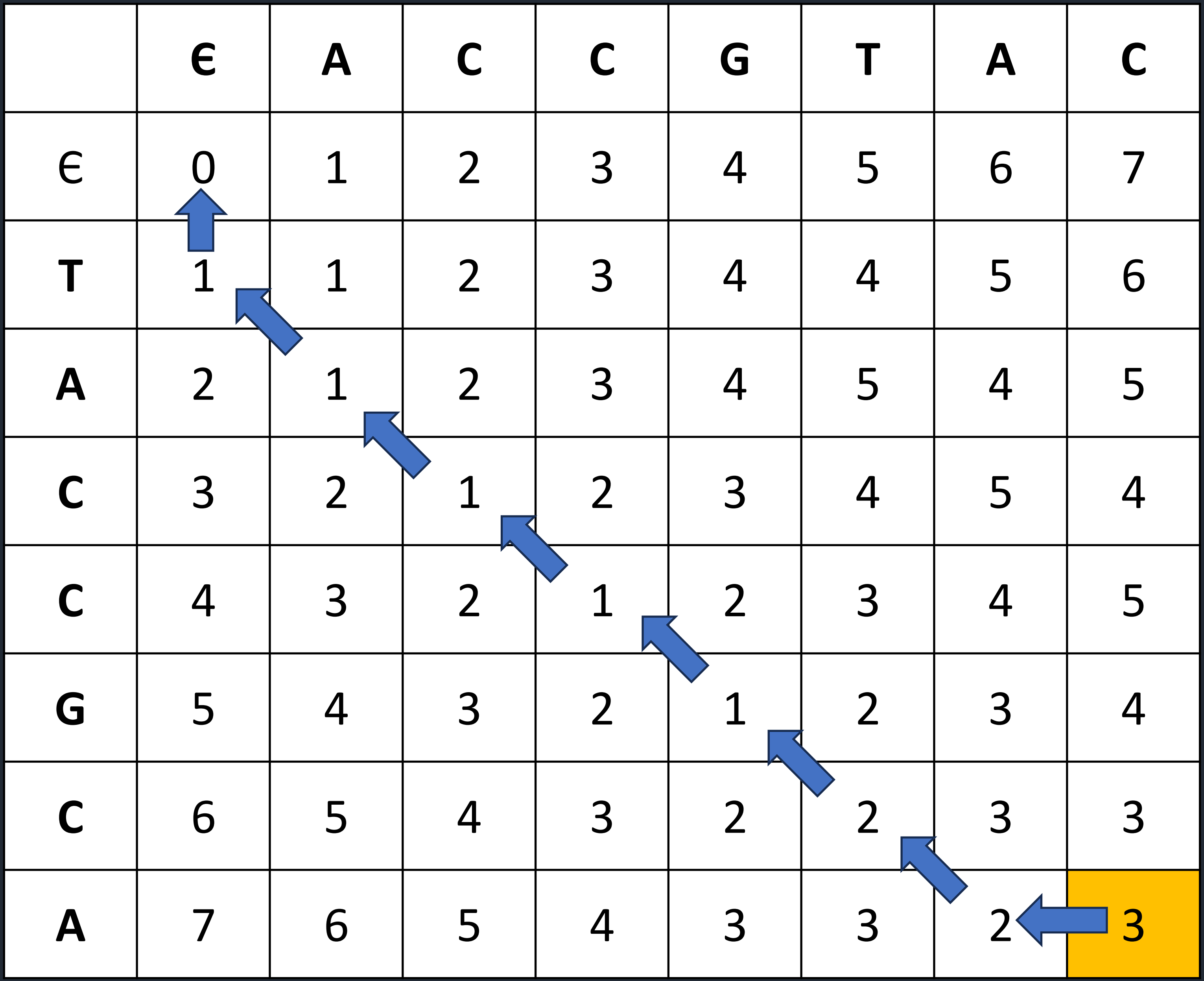


	Є	A	C	C	G	T	A	C
Є	0	1	2	3	4	5	6	7
T	1	1	2	3	4	4	5	6
A	2	1	2	3	4	5	4	5
C	3	2	1	2	3	4	5	4
C	4	3	2	1	2	3	4	5
G	5	4	3	2	1	2	3	4
C	6	5	4	3	2	2	3	3
A	7	6	5	4	3	3	2	3

Step 4:
Traceback

	Є	**A**	**C**	**C**	**G**	**T**	**A**	**C**
Є	0	1	2	3	4	5	6	7
T	↑ 1	1	2	3	4	4	5	6
A	2	↖ 1	2	3	4	5	4	5
C	3	2	↖ 1	2	3	4	5	4
C	4	3	2	↖ 1	2	3	4	5
G	5	4	3	2	↖ 1	2	3	4
C	6	5	4	3	2	↖ 2	3	3
A	7	6	5	4	3	3	↖ 2	← 3

Step 4:
Traceback



	ϵ	A	C	C	G	T	A	C
ϵ	0	1	2	3	4	5	6	7
T	1	1	2	3	4	4	5	6
A	2	1	2	3	4	5	4	5
C	3	2	1	2	3	4	5	4
C	4	3	2	1	2	3	4	5
G	5	4	3	2	1	2	3	4
C	6	5	4	3	2	2	3	3
A	7	6	5	4	3	3	2	3

Transcript:
DMMMMRMI

Transcript:

DMMMMRMI

S1: TACCGCA, S2: ACCGTAC

T	**A**	**C**	**C**	**G**	**C**	**A**	**ϵ**
ϵ	A	C	C	G	T	A	C

Algorithm Sketch

```
1. D(i,0)=i, D(0,j)=j for all i,j

2. Set row=col=0

3. While (row<m or col<n)

       If (row<m) row=row+1
       If (col<n) col=col+1
```

Algorithm Sketch

```
    for (j=row to n)
        Compute Cell(row,j) and
        Parent(row,j) and Transcript(row,j)

    for (i=col+1 to m)
        Compute Cell(i,col) and
        Parent(i,col) and Transcript(row,j)

End While
```

Algorithm Sketch

```
4. Cell=(m,n)
Transcript=ϵ
While (Parent(Cell) is not in zeroth row/column)
        Transcript=Transcript+Transcript(Cell)
        Cell=Parent(Cell)
```

Time and Space Complexity

Space=Size of Table=O(mn)

Time for Table Computation: O(mn)
Time for Traceback: O(m+n)

Space Complexity

Space=Size of Table=O(mn)

Can we do better?