Bioinformatics

Supplementary Slides

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Computer Science Tripos Part II

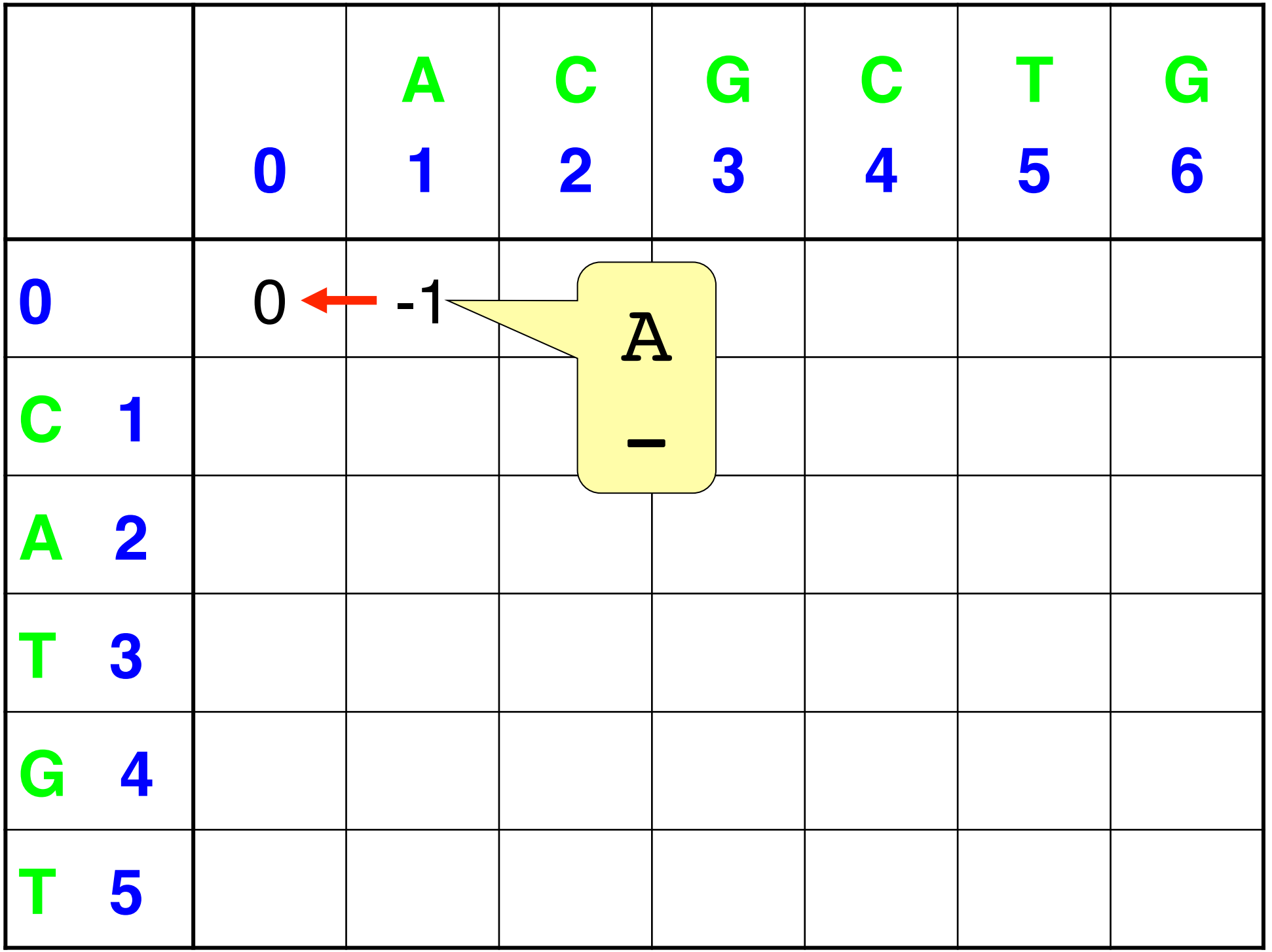
Pietro Lio

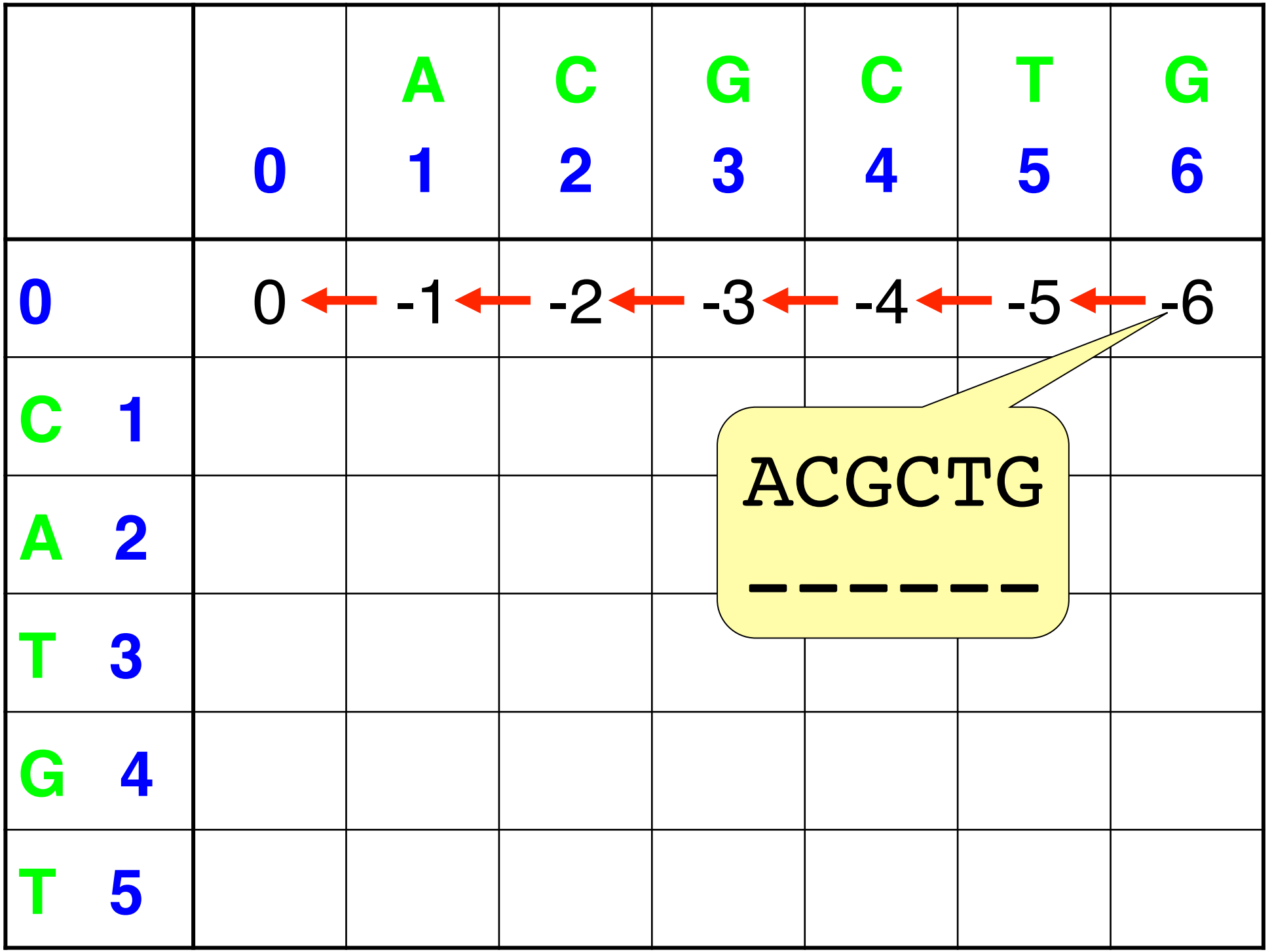
Michaelmas 2012

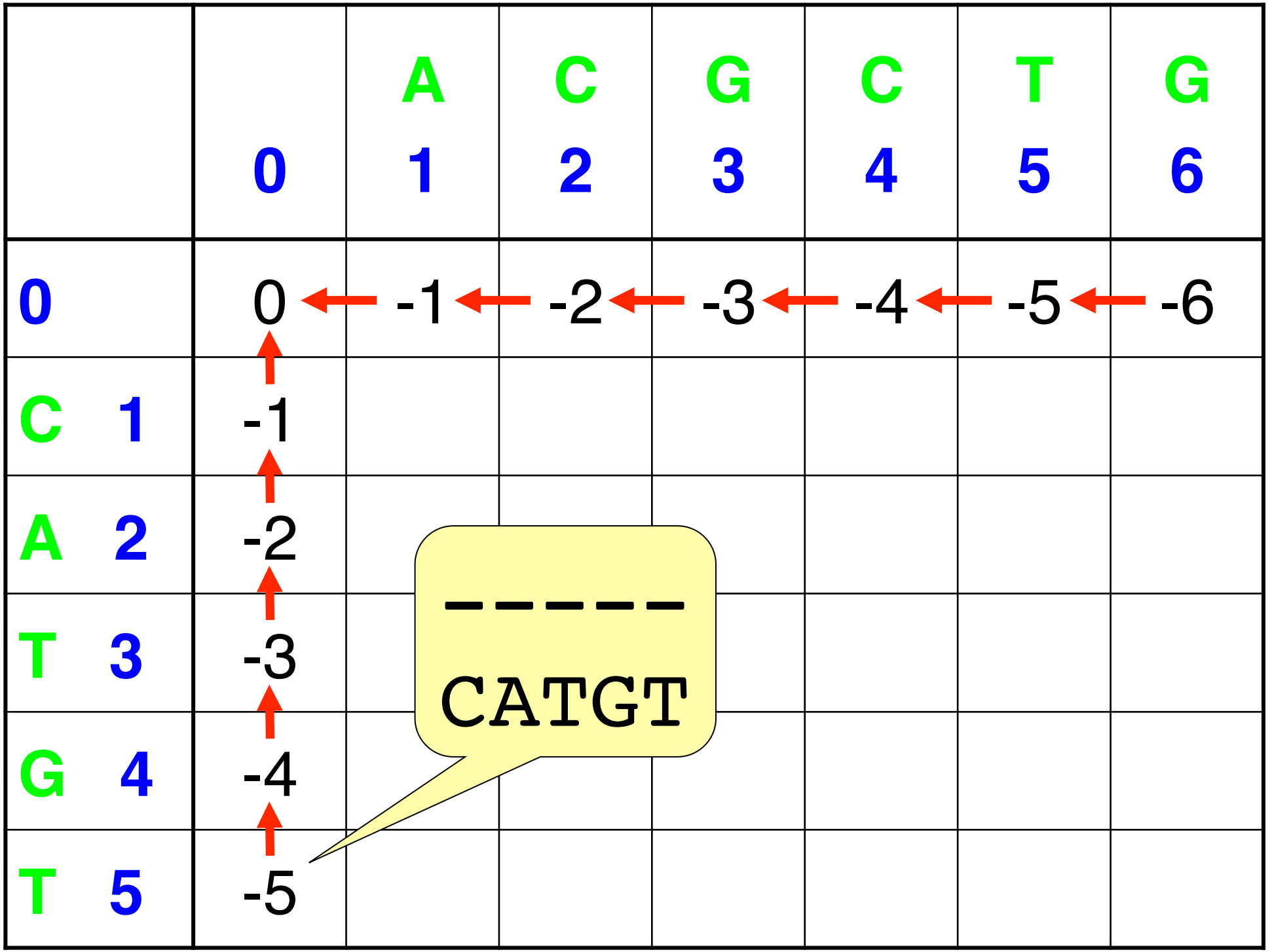
Bioinformatics Examples

Some of the images in these lectures are from several sources that have welcomed instructors to use them for educational purposes (Felsenstein, Moran, Gerstein, Yang, Arkin, Leibler, Batzoglou, Pevzner, Nussinov).

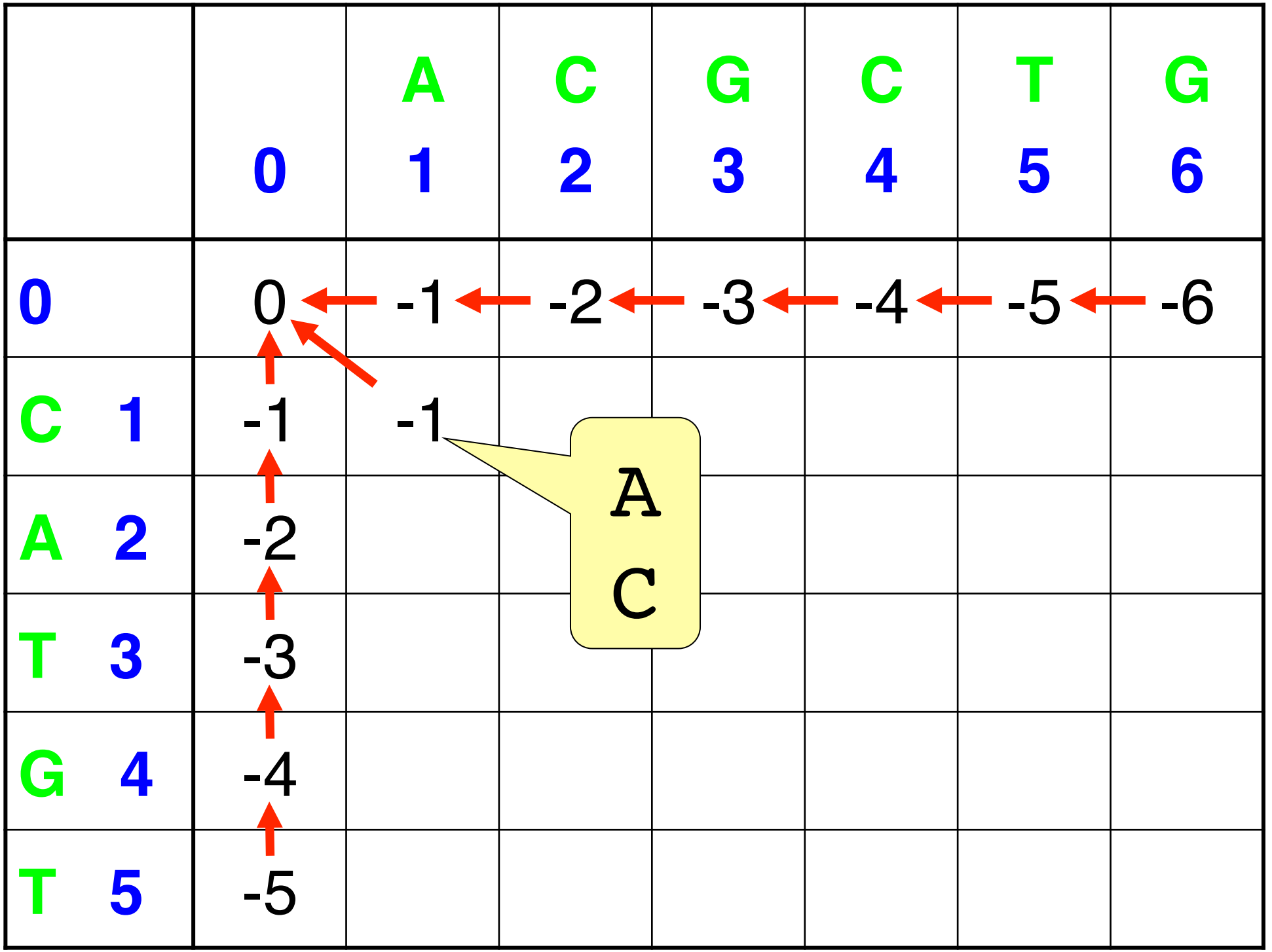
Gap penalty=-1; match=+2; mismatch=-1	0	A 1	C 2	G 3	C 4	T 5	G 6
0	0						
C 1							
A 2							
T 3							
G 4							
T 5							

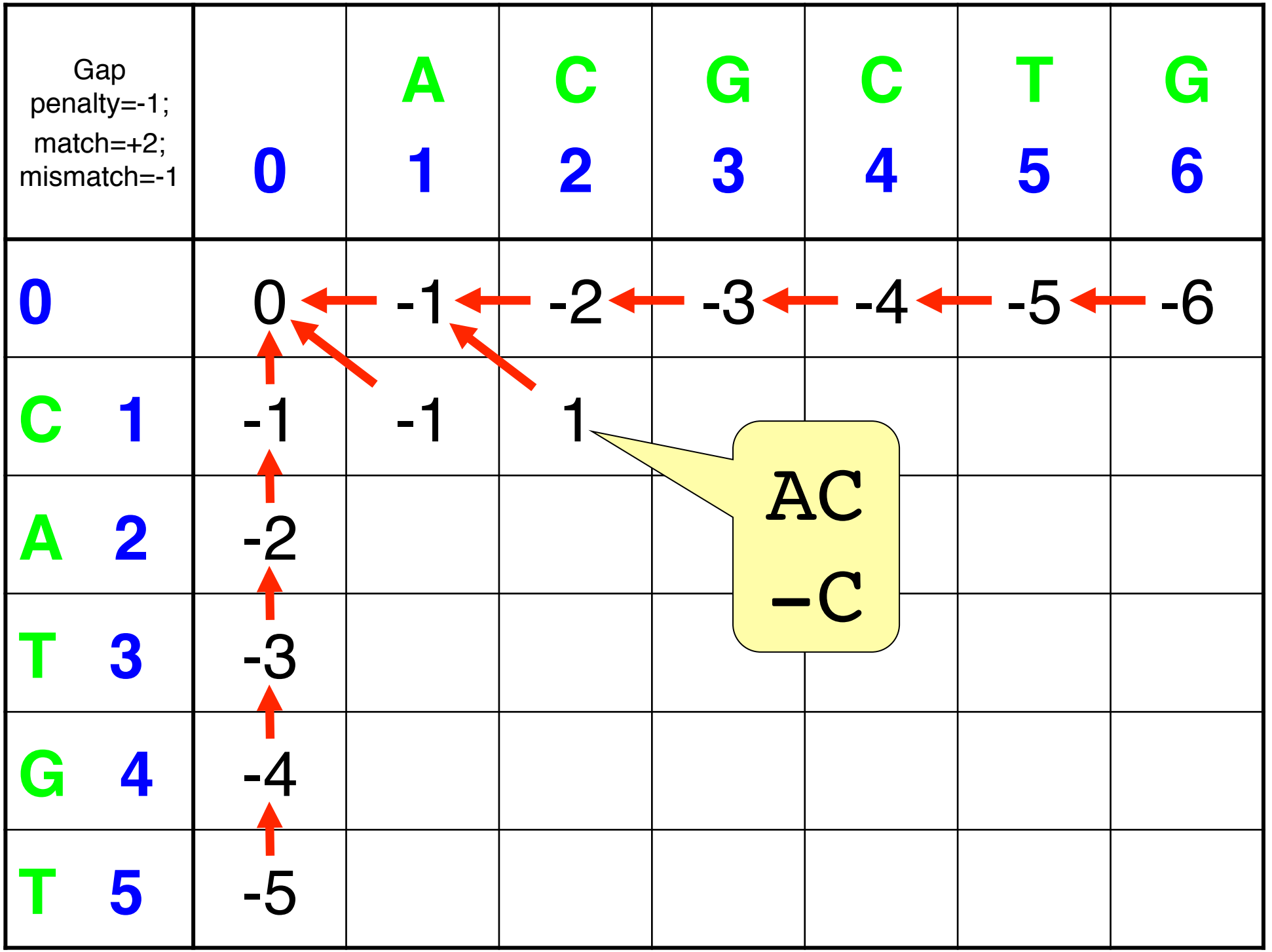

A C G C T G
0 1 2 3 4 5 6
0
C 1
A 2
T 3
G 4
T 5
0
-1
A
–

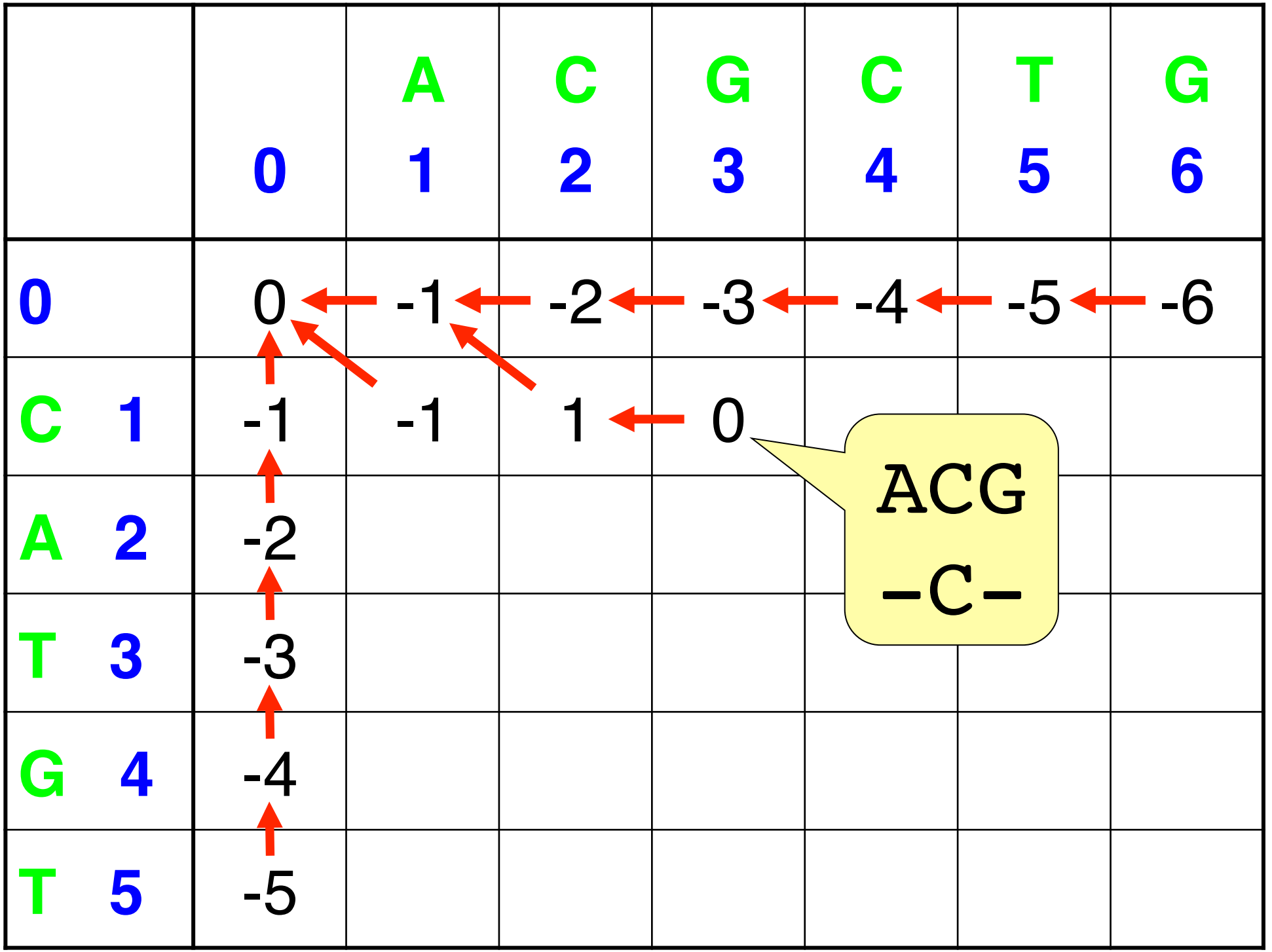

A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1
A 2
T 3
G 4
T 5
ACGCTG

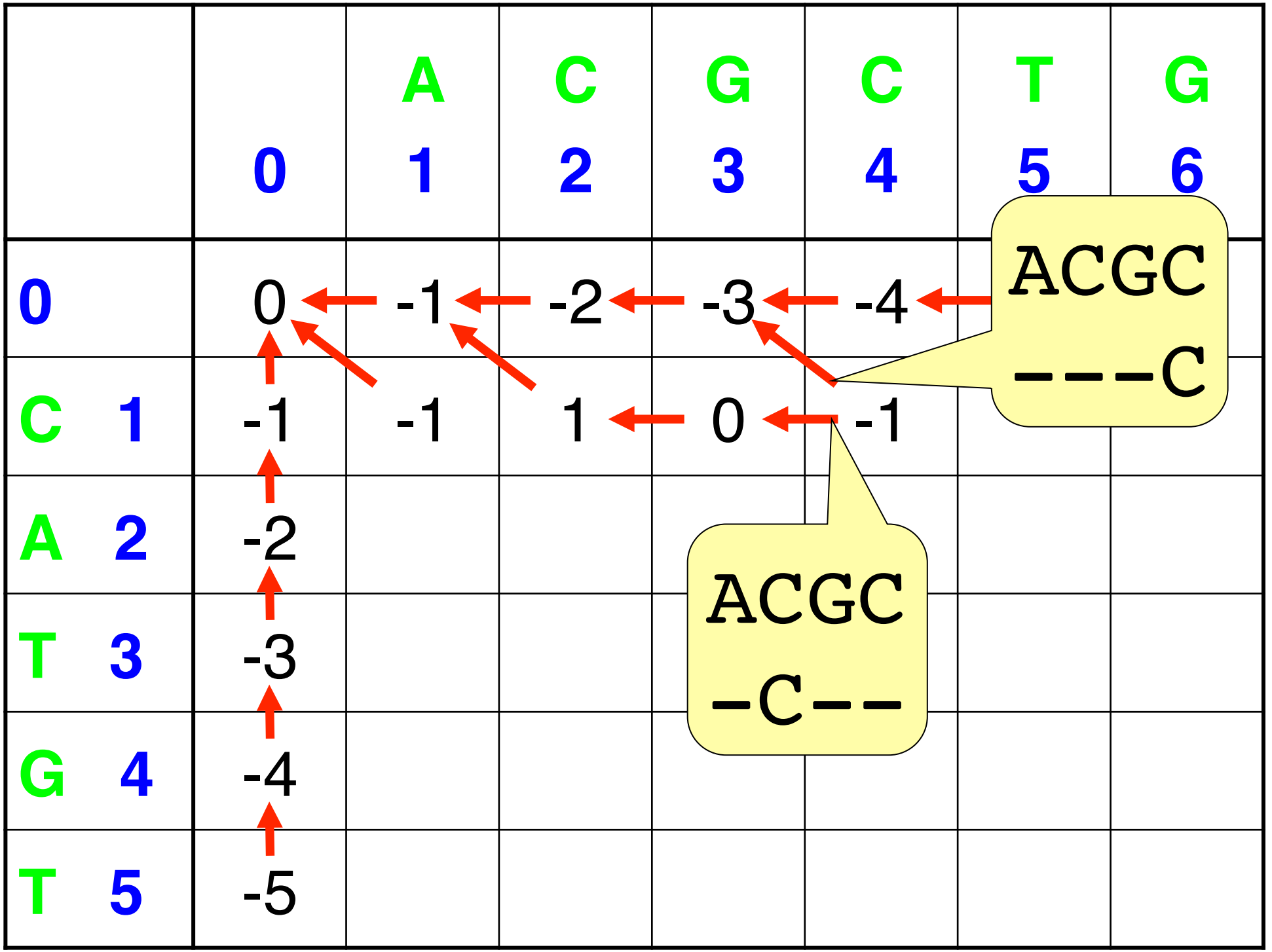

A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1 -1
A 2 -2
T 3 -3
G 4 -4
T 5 -5

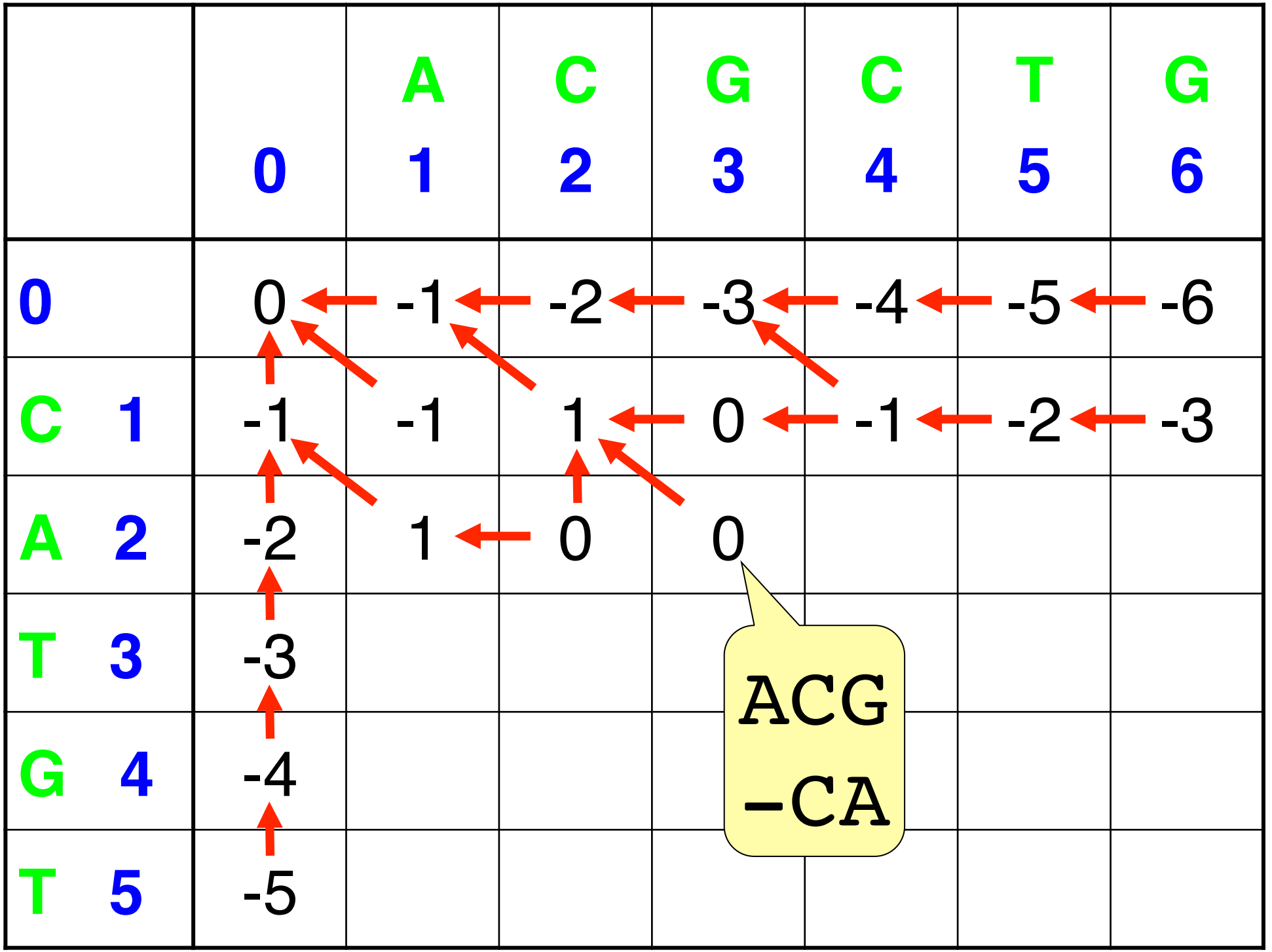
CATGT

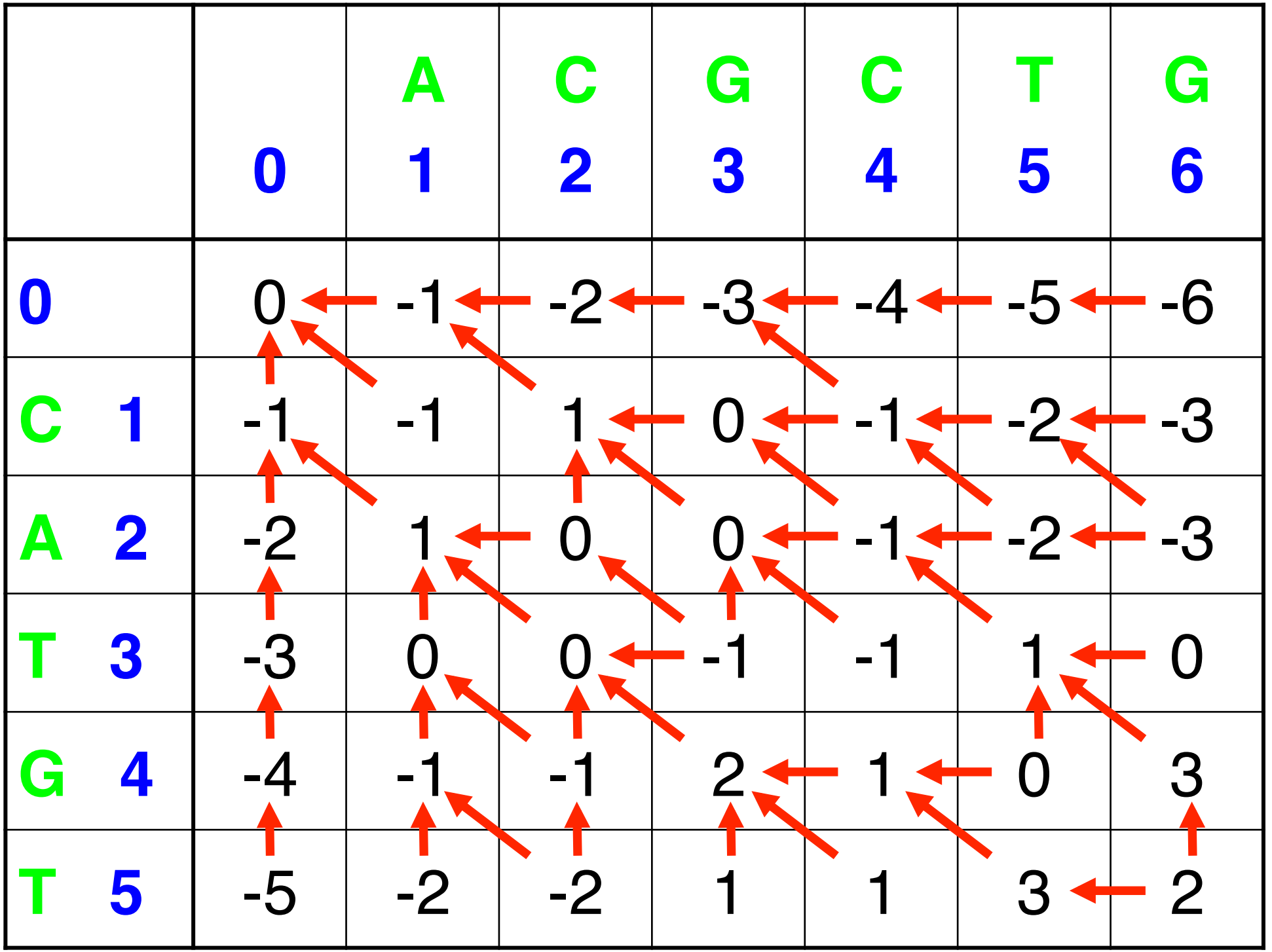

A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1 -1 -1
A 2 -2
T 3 -3
G 4 -4
T 5 -5
A
C

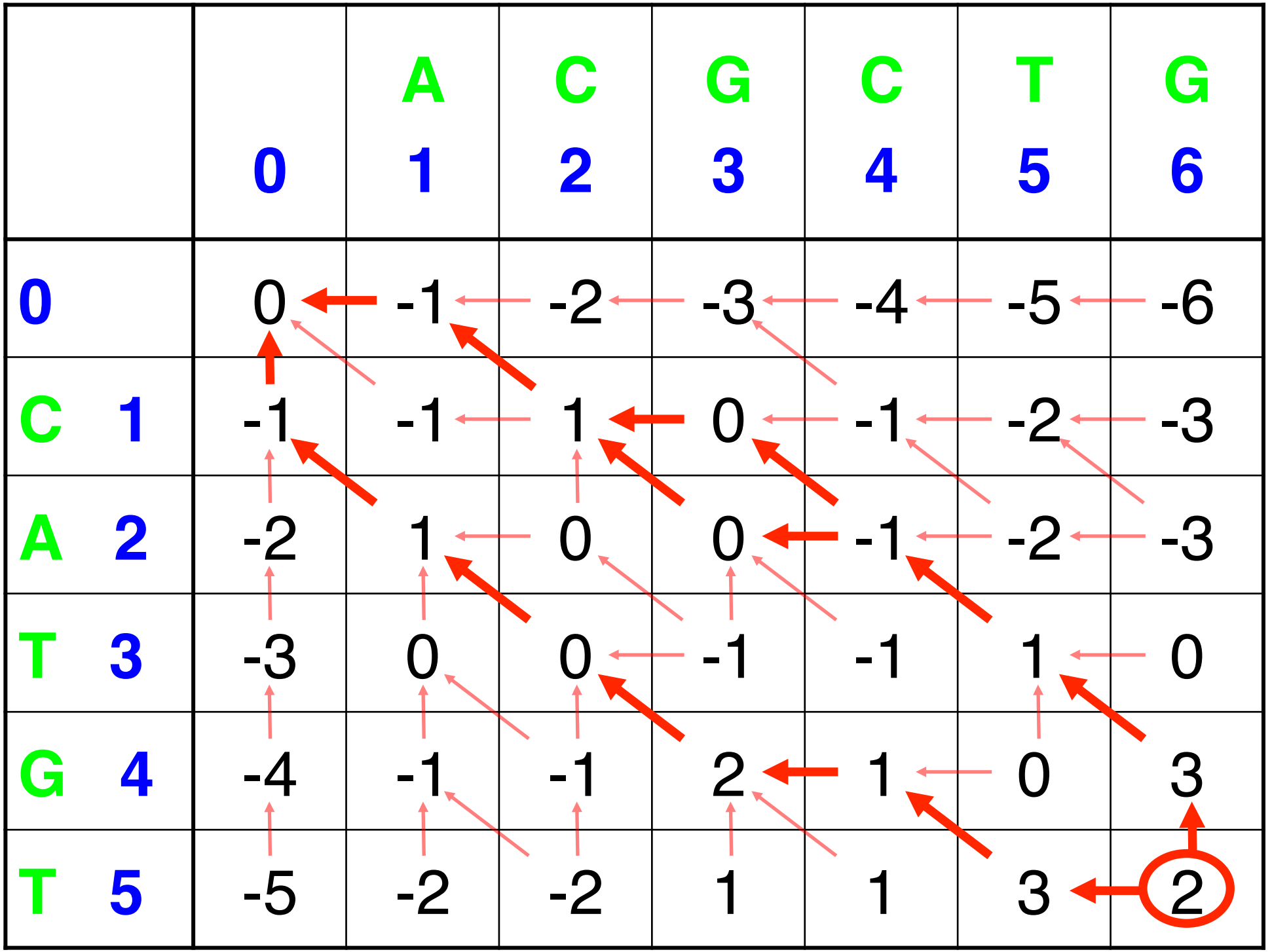

Gap penalty=-1;
match=+2;
mismatch=-1
A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1 -1 -1 1
A 2 -2
T 3 -3
G 4 -4
T 5 -5
AC
–C

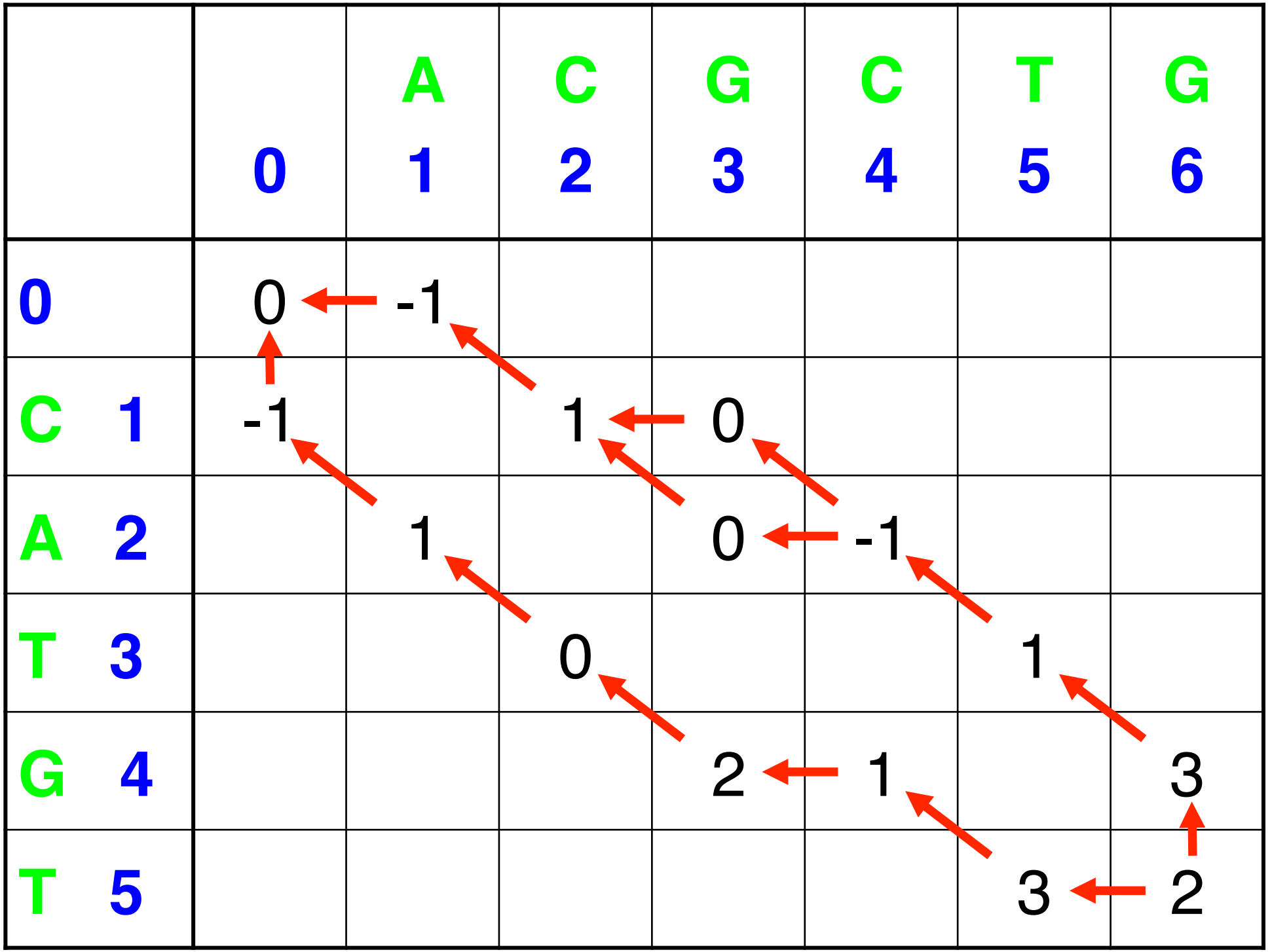

A C G C T G
0 1 2 3 4 5 6
0
0 -1 -2 -3 -4 -5 -6
C 1
-1 -1 1 0
A 2
-2
T 3
-3
G 4
-4
T 5
-5
ACG
–C–

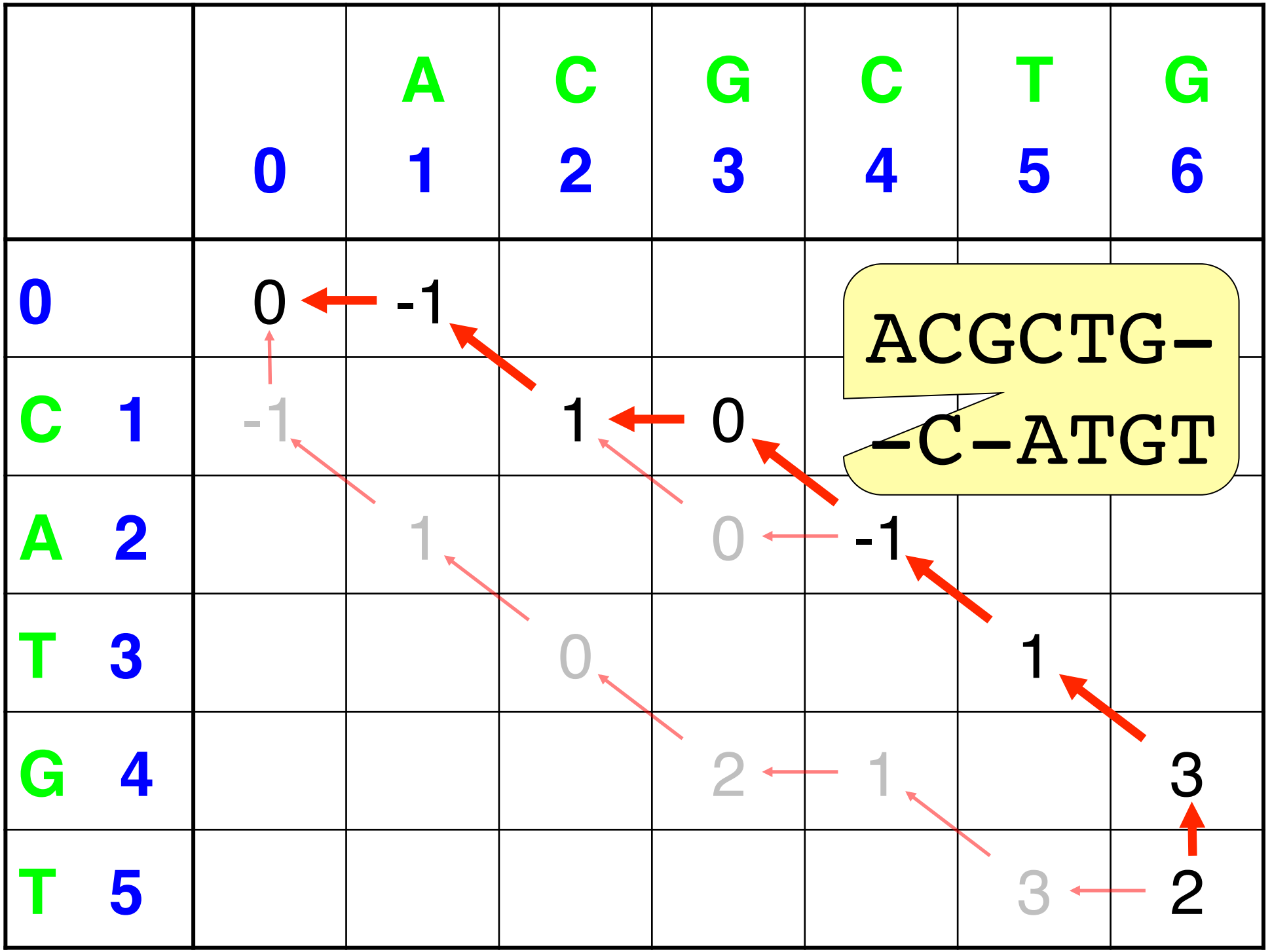

A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4
C 1 -1 -1 1 0 -1
A 2 -2
T 3 -3
G 4 -4
T 5 -5
ACGC
---C
ACGC
-C--

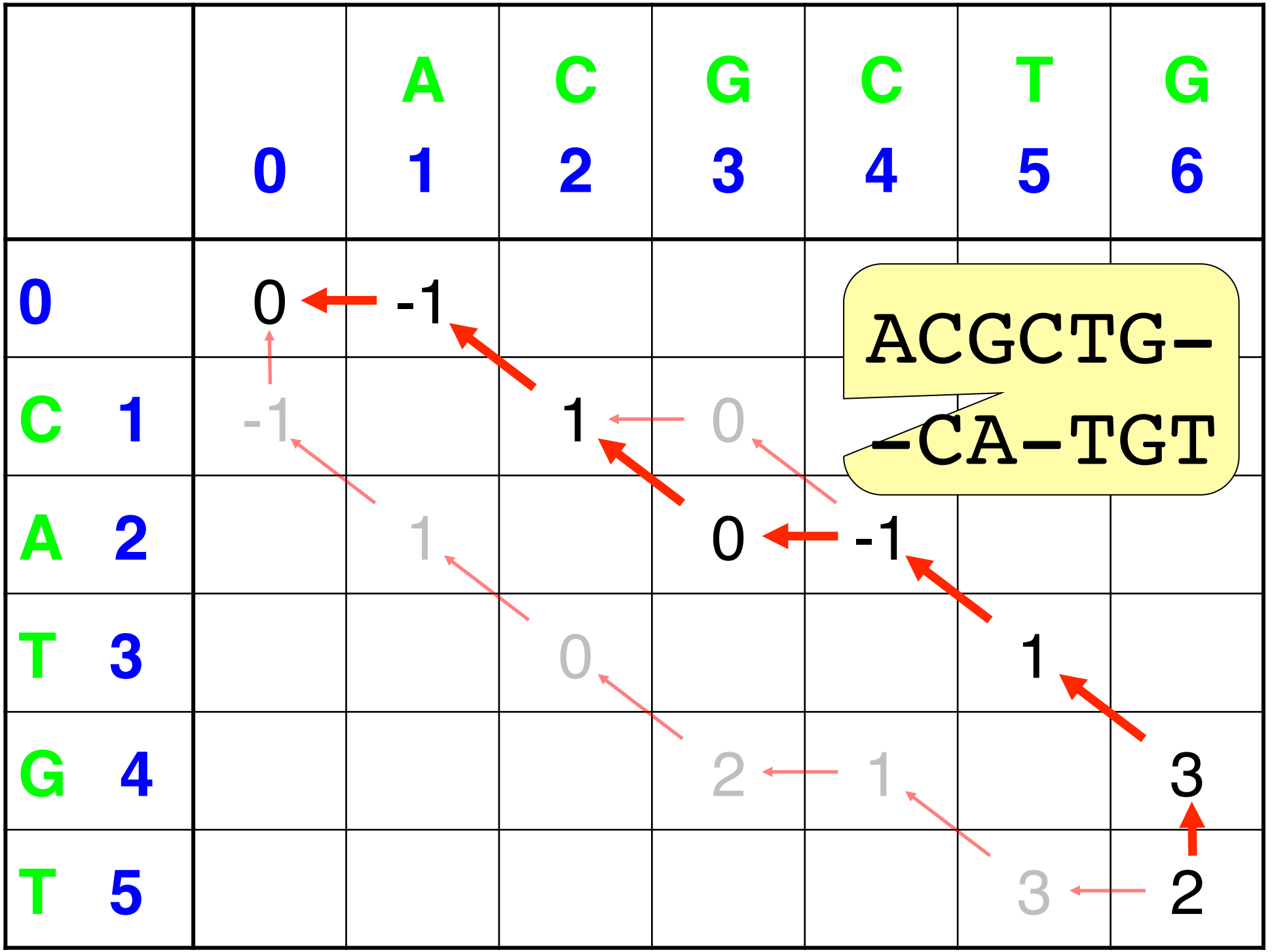

A C G C T G
0 1 2 3 4 5 6
0
0 -1 -2 -3 -4 -5 -6
C 1
-1 -1 1 0 -1 -2 -3
A 2
-2 1 0 0
T 3
-3
G 4
-4
T 5
-5
ACG
–CA

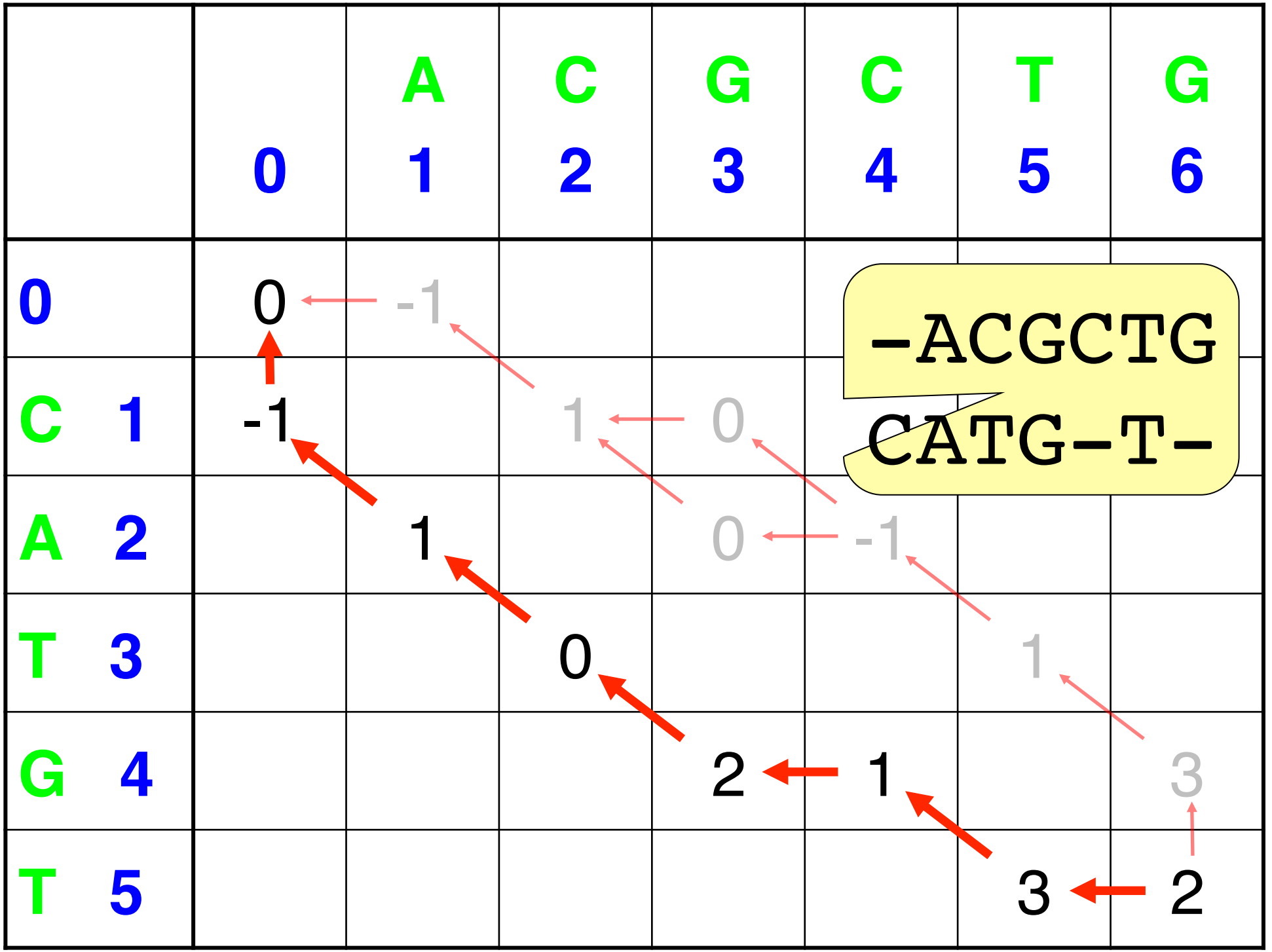

A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1 -1 -1 1 0 -1 -2 -3
A 2 -2 1 0 0 -1 -2 -3
T 3 -3 0 0 -1 -1 1 0
G 4 -4 -1 -1 2 1 0 3
T 5 -5 -2 -2 1 1 3 2


A C G C T G
0 1 2 3 4 5 6
0 0 -1 -2 -3 -4 -5 -6
C 1 -1 -1 1 0 -1 -2 -3
A 2 -2 1 0 0 -1 -2 -3
T 3 -3 0 0 -1 -1 1 0
G 4 -4 -1 -1 2 1 0 3
T 5 -5 -2 -2 1 1 3 2


A C G C T G
0 1 2 3 4 5 6
0
C 1
A 2
T 3
G 4
T 5
0 -1
-1 1 0
1 0 -1
0 1
2 1 3
3 2


A C G C T G
0 1 2 3 4 5 6
0
C 1
A 2
T 3
G 4
T 5
ACGCTG-
-C-ATGT


A C G C T G
0 1 2 3 4 5 6
0
C 1
A 2
T 3
G 4
T 5
ACGCTG-
-CA-TGT


A C G C T G
0 1 2 3 4 5 6
0
C 1
A 2
T 3
G 4
T 5
-ACGCTG
CATG-T-

Local Alignment Example

y = TAATA
x = TACTAA

y \ x		A	T	C	T	A	A
	0	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
T 1	0						
A 2	0						
A 3	0						
T 4	0						
A 5	0						

Local Alignment Example

y = TAATA
x = TACTAA

		T	**A**	**C**	**T**	**A**	**A**
	0	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
T 1	0	1	0	0	1	0	0
A 2	0	0	2	0	0	2	1
A 3	0						
T 4	0						
A 5	0						

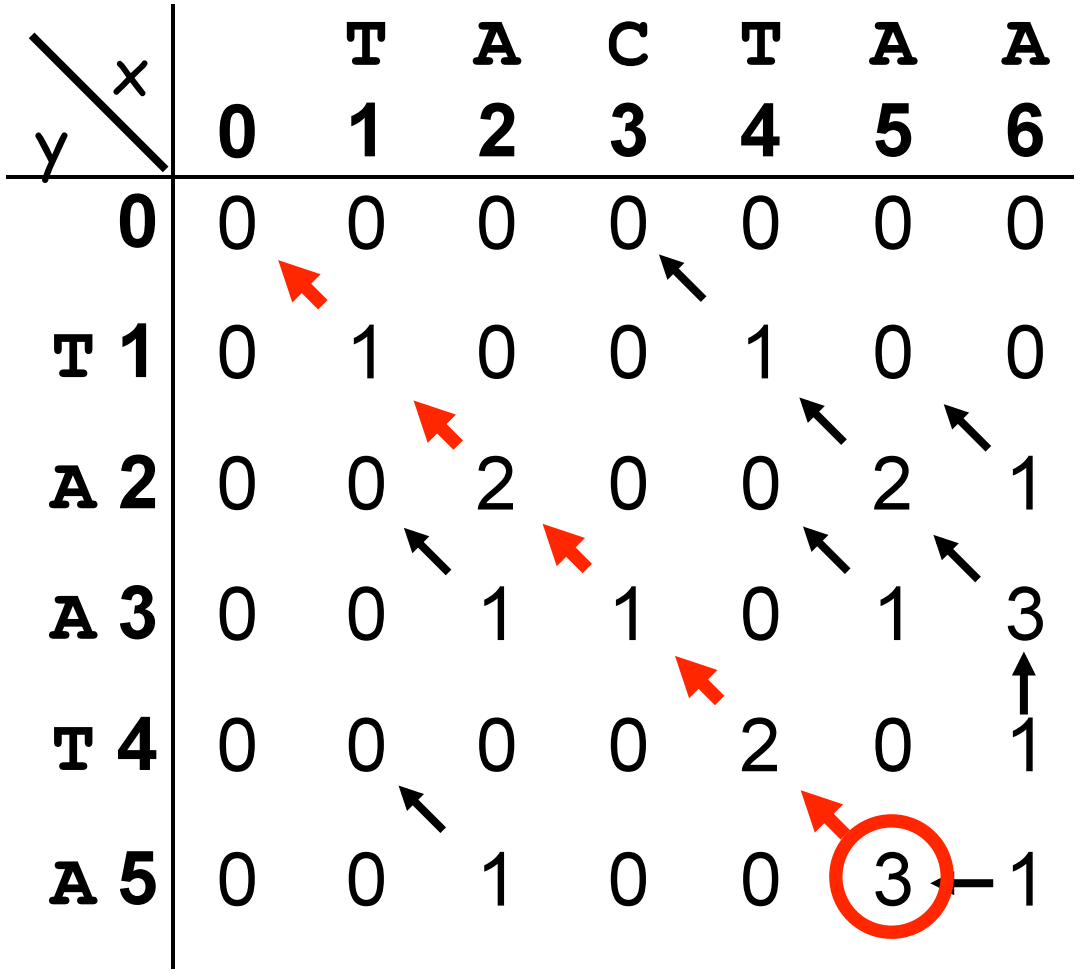
Local Alignment Example

y = TAATA
x = TACTAA

		T	**A**	**C**	**T**	**A**	**A**
	0	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
T **1**	0	1	0	0	1	0	0
A **2**	0	0	2	0	0	2	1
A **3**	0	0	1	1	0	1	3
T **4**	0	0	0	0	2	0	1
A **5**	0	0	1	0	0	3	←1

Local Alignment Example

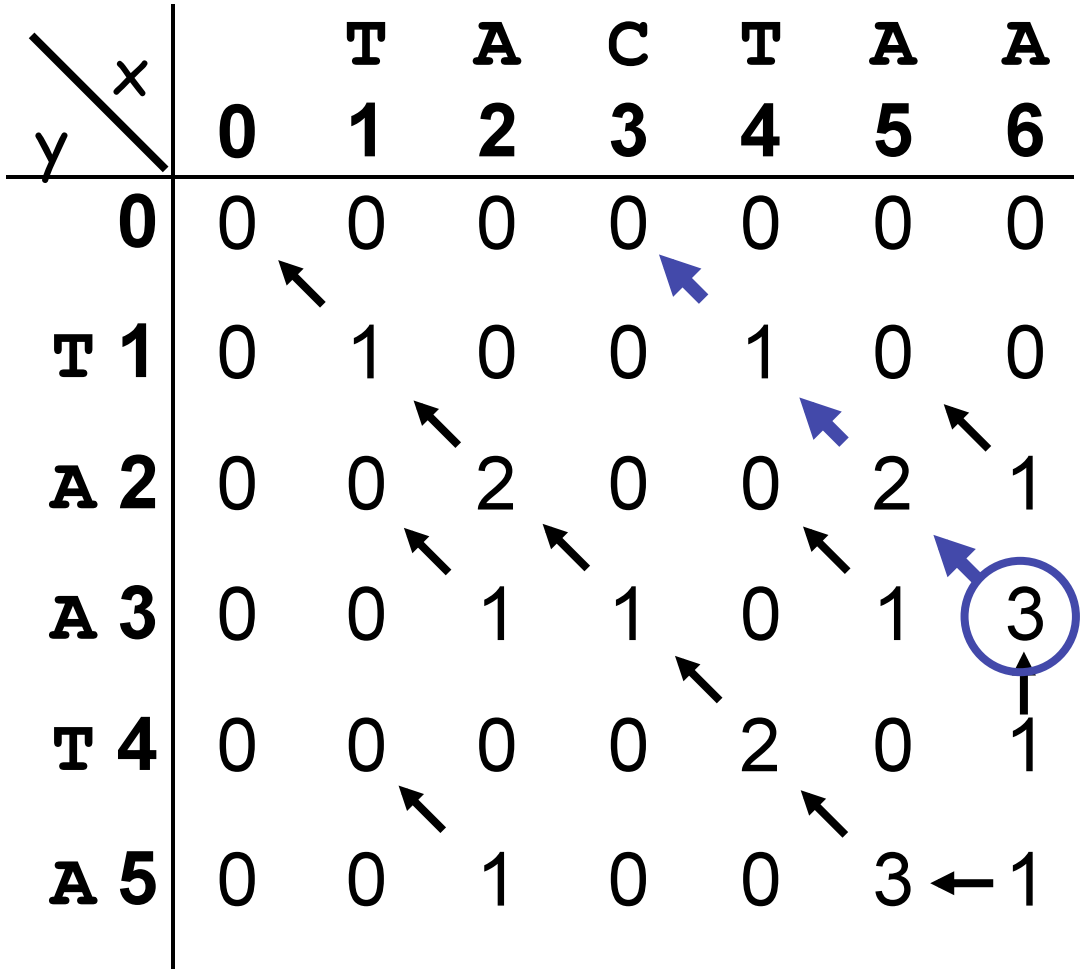
y = TAATA
x = TACTAA



Local Alignment Example

$y =$ TAATA

$x =$ TACTAA

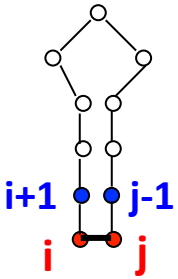


RNA Secondary Structure

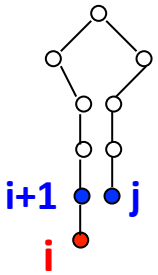
- Secondary Structure :
  - Set of paired positions on interval $[i,j]$
  - This tells which bases are paired in the subsequence from x_i to x_j
- Every optimal structure can be built by extending optimal substructures.
- Suppose we know all optimal substructures of length less than $j\text{-}i+1$.

  The optimal substructure for $[i,j]$ must be formed in one of four ways:
  1. i,j paired
  2. i unpaired
  3. j unpaired
  4. combining two substructures

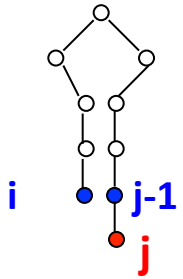
  Note that each of these consists of extending or joining substructures of length less than $j\text{-}i+1$.



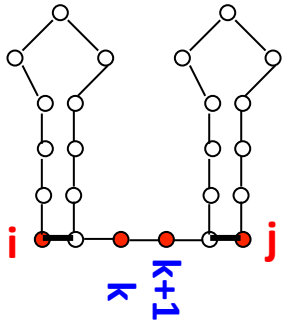
i,j pair



i unpaired



j unpaired



bifurcation

The Nussinov Folding Algorithm

Example: GGGAAAUCC

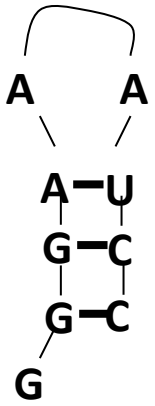
$\gamma(i,j)$ is the maximum number of base pairs in segment $[i,j]$

$$Initialisation\ \gamma(i,i-1) = 0\ \&\ \gamma(i,i) = 0$$

Starting with all subsequences of length 2, to length L:

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \delta(i,j) \\ \max_{i<k<j}[\gamma(i,k) + \gamma(k+1,j)] \end{cases}$$

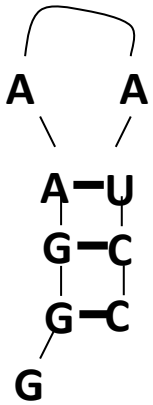
Where $\delta(i,j) = 1$ if x_i and x_j are a complementary base pair, and $\delta(i,j) = 0$, otherwise.



i \ j	G	G	G	A	A	A	U	C	C
G	0								
G	0	0							
G		0	0						
A			0	0					
A				0	0				
U					0	0			
C						0	0		
C							0	0	
								0	0

Nussinov Folding Algorithm: After scores for subsequences of length 2

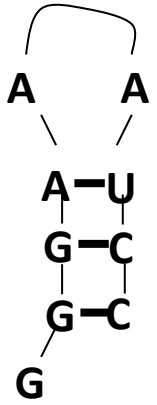
$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \delta(i,j) \\ \max_{i<k<j}[\gamma(i,k) + \gamma(k+1,j)] \end{cases}$$



i \ j	G	G	G	A	A	A	U	C	C
G	0	**0**							
G	0	0	**0**						
G		0	0	**0**					
A			0	0	**0**				
A				0	0	**0**			
U					0	0	**1**		
C						0	0	**0**	
							0	0	**0**
								0	0

Nussinov Folding Algorithm: After scores for subsequences of length 3

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1)+\delta(i,j) \\ \max_{i<k<j}[\gamma(i,k)+\gamma(k+1,j)] \end{cases}$$



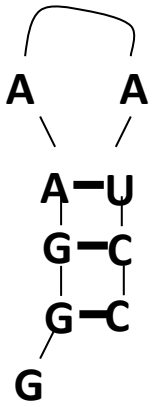
j →

i ↓

		G	G	G	A	A	A	U	C	C
	0	0	**0**						
G	0	0	0	**0**					
G		0	0	0	**0**				
G			0	0	0	**0**			
A				0	0	0	**1**		
A					0	0	1	**0**	
U						0	0	0	**0**
C							0	0	0
								0	0

Nussinov Folding Algorithm
After scores for subsequences of length 4

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \delta(i,j) \\ \max_{i<k<j}[\gamma(i,k) + \gamma(k+1,j)] \end{cases}$$



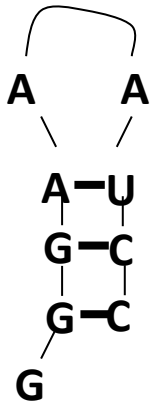
j →

i		G	G	G	A	A	A	U	C	C
		0	0	0	**0**					
G		0	0	0	0	**0**				
G			0	0	0	0	**0**			
G				0	0	0	0	**1**		
A					0	0	**0**	**1**	**1**	
A						0	0	1	1	**1**
U							0	0	0	0
C								0	0	0
									0	0

Two optimal substructures for same subsequence

Nussinov Folding Algorithm
After scores for subsequences of length 5

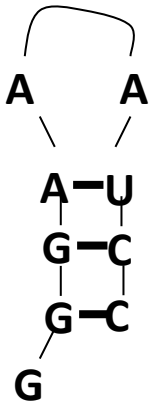
$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1)+\delta(i,j) \\ \max_{i<k<j}[\gamma(i,k)+\gamma(k+1,j)] \end{cases}$$



i \ j	G	G	G	A	A	A	U	C	C
G	0	0	0	0	**0**				
G	0	0	0	0	0	**0**			
G		0	0	0	0	0	**1**		
A			0	0	0	0	1	**1**	
A				0	0	0	1	1	**1**
U					0	0	1	1	1
C						0	0	0	0
							0	0	0
								0	0

Nussinov Folding Algorithm
After scores for subsequences of length 6

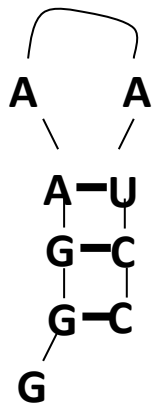
$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \delta(i,j) \\ \max_{i<k<j}[\gamma(i,k) + \gamma(k+1,j)] \end{cases}$$



		G	G	G	A	A	A	U	C	C
		j →								
G	i ↓	0	0	0	0	0	**0**			
G		0	0	0	0	0	0	**1**		
G			0	0	0	0	0	1	**2**	
A				0	0	0	0	1	1	**1**
A					0	0	0	1	1	1
U						0	0	1	1	1
C							0	0	0	0
								0	0	0
									0	0

Nussinov Folding Algorithm
After scores for subsequences of length 7

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1)+\delta(i,j) \\ \max_{i<k<j}[\gamma(i,k)+\gamma(k+1,j)] \end{cases}$$

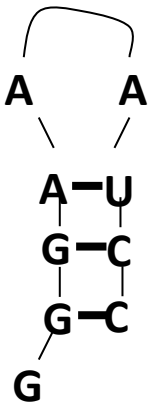


j →

i		G	G	G	A	A	A	U	C	C
G		0	0	0	0	0	0	**1**		
G		0	0	0	0	0	0	1	**2**	
G			0	0	0	0	0	1	2	**2**
A				0	0	0	0	1	1	1
A					0	0	0	1	1	1
U						0	0	1	1	1
C							0	0	0	0
								0	0	0
									0	0

Nussinov Folding Algorithm
After scores for subsequences of length 8

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1)+\delta(i,j) \\ \max_{i<k<j}[\gamma(i,k)+\gamma(k+1,j)] \end{cases}$$



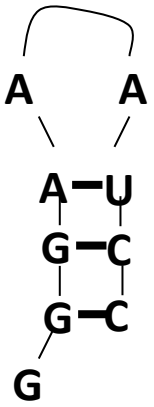
j →

i ↓

	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	1	**2**	
G	0	0	0	0	0	0	1	2	**3**
G		0	0	0	0	0	1	2	2
A			0	0	0	0	1	1	1
A				0	0	0	1	1	1
A					0	0	1	1	1
U						0	0	0	0
C							0	0	0
								0	0

Nussinov Folding Algorithm
After scores for subsequences of length 9

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1)+\delta(i,j) \\ \max_{i<k<j}[\gamma(i,k)+\gamma(k+1,j)] \end{cases}$$

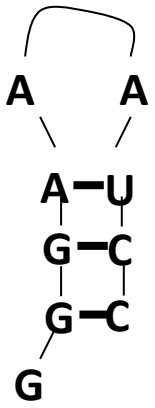


j →

i ↓

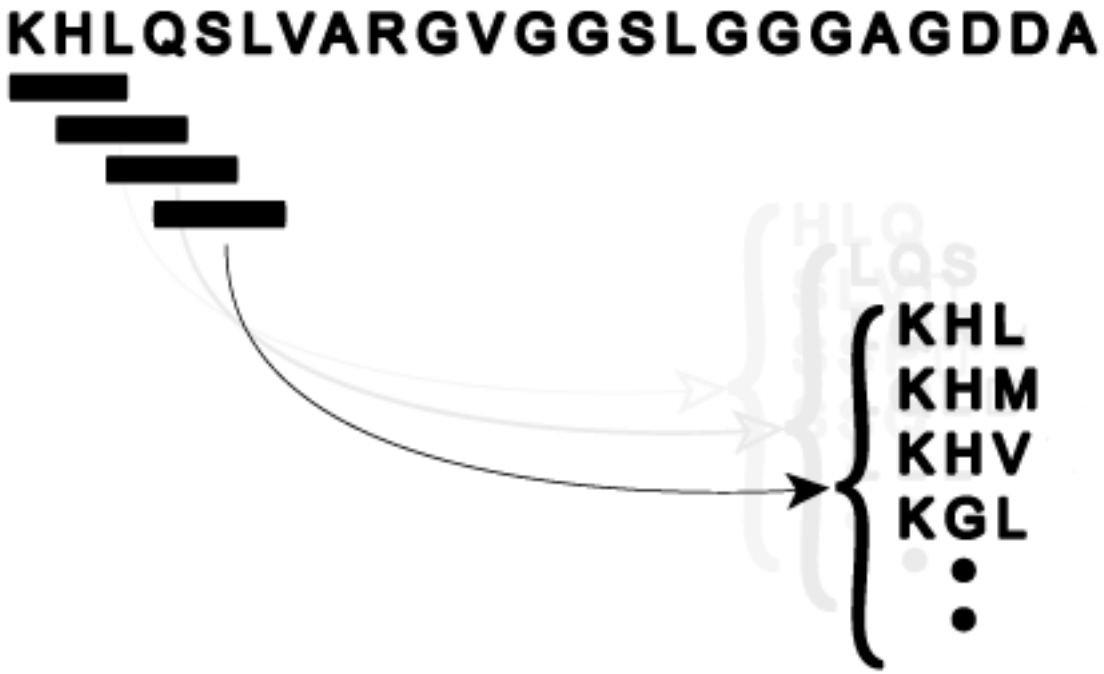
	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	1	2	**3**
G	0	0	0	0	0	0	1	2	3
G		0	0	0	0	0	1	2	2
A			0	0	0	0	1	1	1
A				0	0	0	1	1	1
U					0	0	1	1	1
C						0	0	0	0
							0	0	0
								0	0

Nussinov Folding Algorithm Traceback

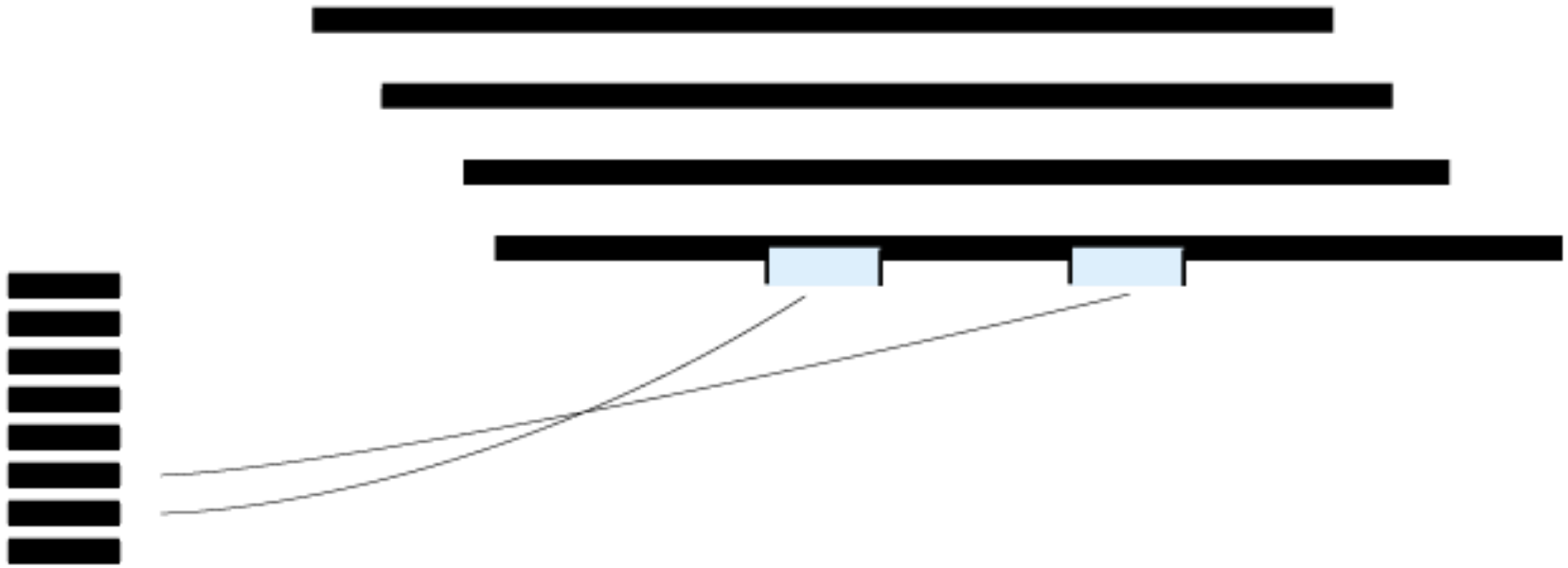


i \ j	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	1	2	3
G	0	0	0	0	0	0	1	2	3
G		0	0	0	0	0	1	2	2
A			0	0	0	0	1	1	1
A				0	0	0	1	1	1
U					0	0	1	1	1
C						0	0	0	0
							0	0	0
								0	0

BLAST



BLAST

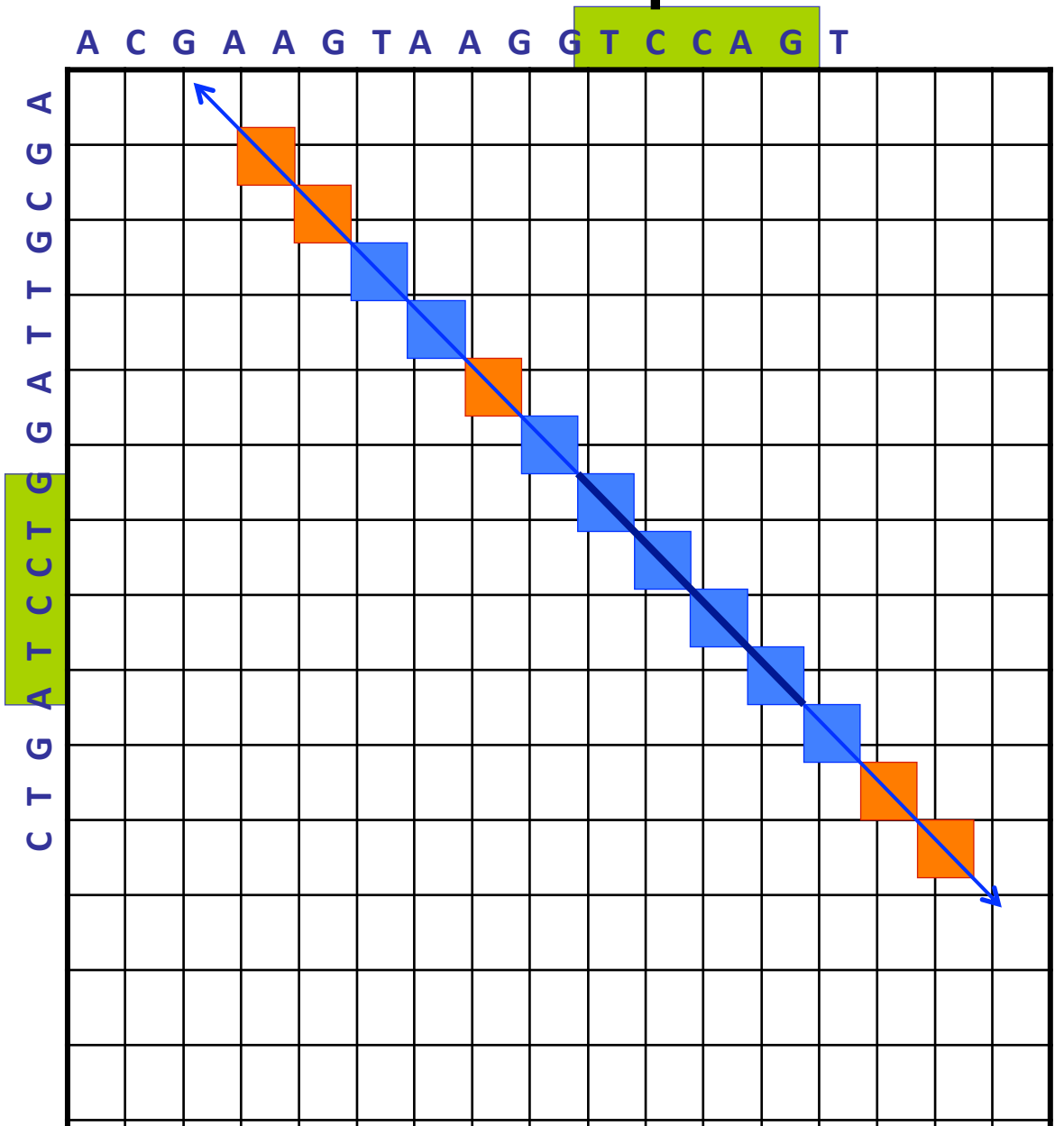


BLAST

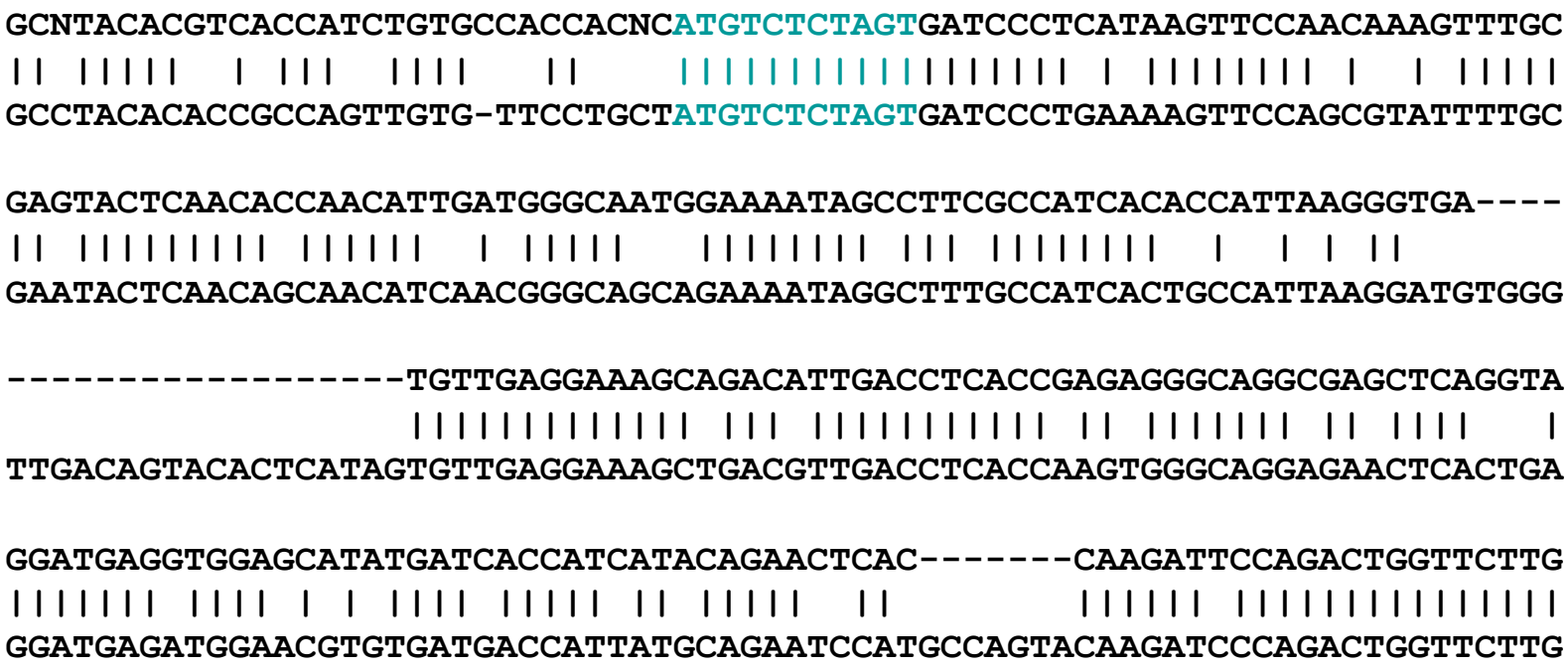
Original BLAST: Example

- $w = 4$
- Exact keyword match of **GGTC**
- Extend diagonals with mismatches until score is under 50%
- Output result

GTAAGGTCC
GTTAGGTCC

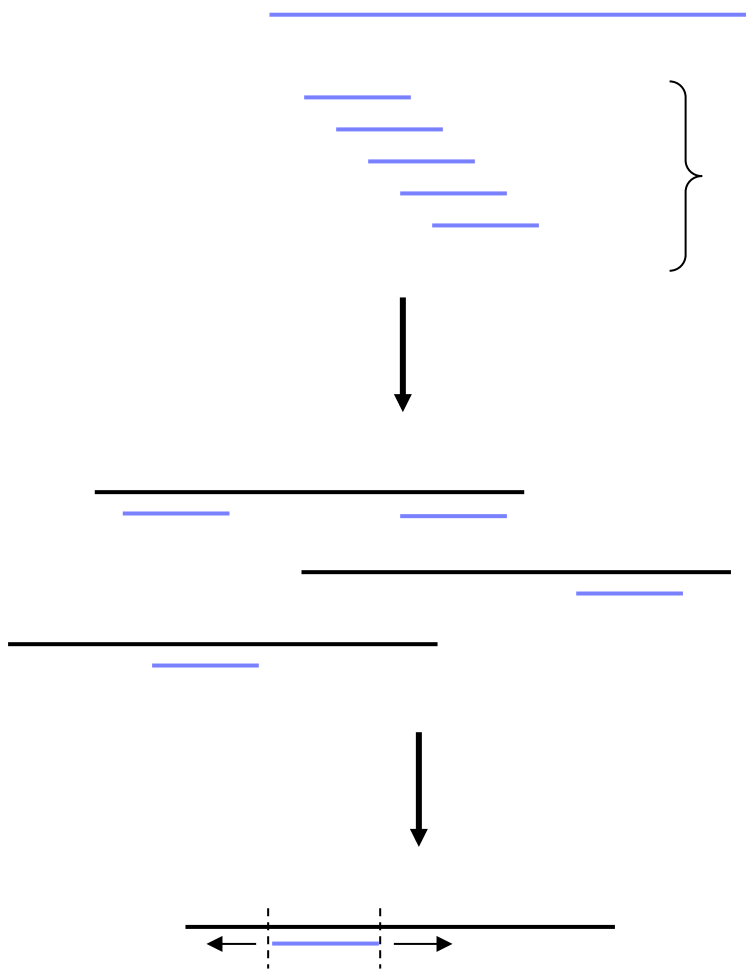


BLAST finds a “hit” and then extends



Seed match = hit

BLAST



Query Sequence

“words” (subsequences of the query sequence)

Query words are compared to the database (target sequences) and exact matches identified

For each word match, alignment is extended in both directions to find alignments that score greater than some threshold

BLAST: example of missing a target

- Fail:

```
GAGTACTCAACACCAACATTAGTGGGCAATGGAAAAT
||  |||||||||  ||||||  |  ||||||    ||||||
GAATACTCAACAGCAACATCAATGGGCAGCAGAAAAT
```

- Dilemma
  - Sensitivity – needs shorter seeds
    - the success rate of finding a homology
  - Speed – needs longer seeds
    - Mega-BLAST uses seeds of length 28.

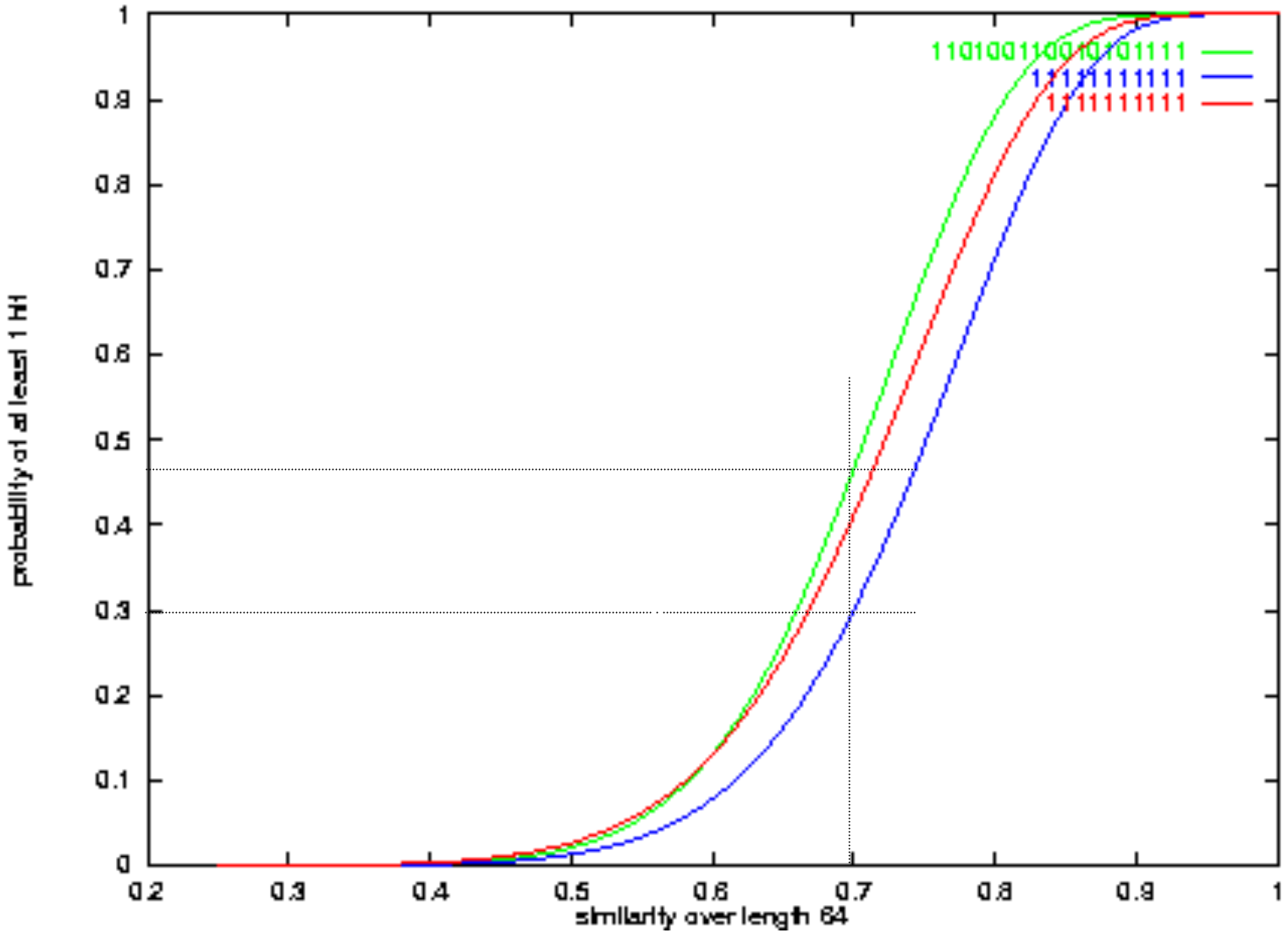
PatternHunter uses “spaced seeds”

- 1110100101001101111 (called a model)
  - Eleven required matches (weight=11)
  - Seven “don’t care” positions

```
GAGTACTCAACACCAACATTAGTGGCAATGGAAAAT…
||  ||||||||||  |||||  ||  ||||||    ||||||
GAATACTCAACAGCAACACTAATGGCAGCAGAAAAT…
        1110100101001101111
```

- Hit = all the required matches are satisfied.
- BLAST seed model = 11111111111

Simulated sensitivity curves



PH2:

(homology identity = 0.7, homology length=64)

11101100101101010111,

11110001000100110101111,

11001101001010001101111,

11101000111110010001101,

……

……

A multiple alignment

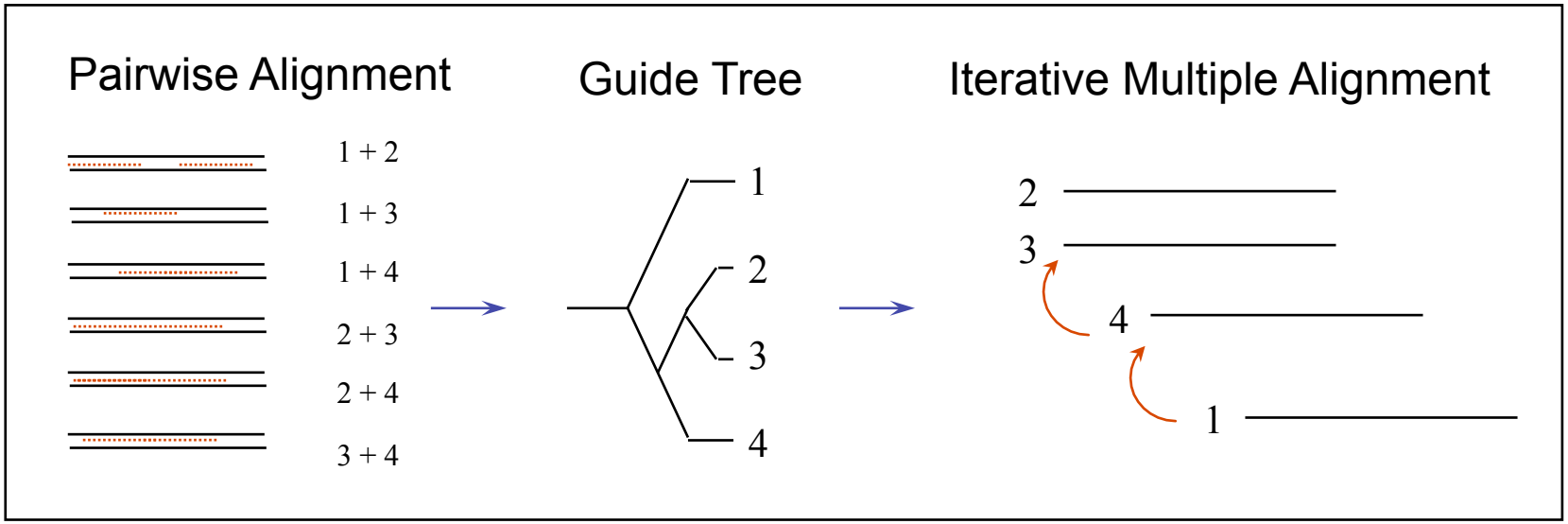
```
                        10        20        30        40        50        60
                         .         .         .         .         .         .
Hbb_Human.pep   --------VHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDLST
Hbb_Horse.pep   --------VQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDLSN
Hba_Human.pep   ---------VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLS--
Hba_Horse.pep   ---------VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLS--
Myg_Phyca.pep   ---------VLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLKT
Glb5_Petma.pep  PIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLTT
Lgb2_Luplu.pep  --------GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE
                         *.      .   *                       * .   *

Hbb_Human.pep   PDAVMGNPKVKAHGKKVLGAFSDGLAHLD-----NLKGTFATLSELHCDKLHVDPENFRL
Hbb_Horse.pep   PGAVMGNPKVKAHGKKVLHSFGEGVHHLD-----NLKGTFAALSELHCDKLHVDPENFRL
Hba_Human.pep   ----HGSAQVKGHGKKVADALTNAVAHVD-----DMPNALSALSDLHAHKLRVDPVNFKL
Hba_Horse.pep   ----HGSAQVKAHGKKVGDALTLAVGHLD-----DLPGALSNLSDLHAHKLRVDPVNFKL
Myg_Phyca.pep   EAEMKASEDLKKHGVTVLTALGAILKKKG-----HHEAELKPLAQSHATKHKIPIKYLEF
Glb5_Petma.pep  ADQLKKSADVRWHAERIINAVNDAVASMDDT--EKMSMKLRDLSGKHAKSFQVDPQYFKV
Lgb2_Luplu.pep  VP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-VADAHFPV
                        .. *   .                       *   *     .

Hbb_Human.pep   LGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------
Hbb_Horse.pep   LGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH------
Hba_Human.pep   LSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------
Hba_Horse.pep   LSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSKYR------
Myg_Phyca.pep   ISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
Glb5_Petma.pep  LAAVIADTVAAG---------DAGFEKLMSMICILLRSAY-------
Lgb2_Luplu.pep  VKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA---
                .   .   .                       .
```

Progressive Alignment

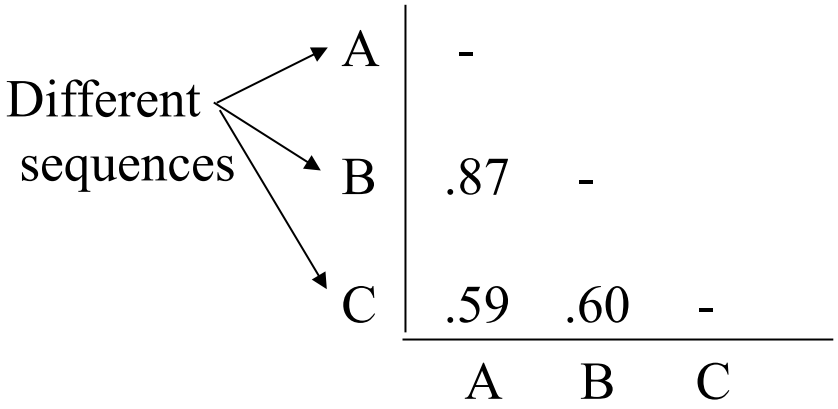
Principle:



1 → **2** → **3**

Step 1-pairwise alignments

Compare each sequence with each other and calculate a Similarity matrix.

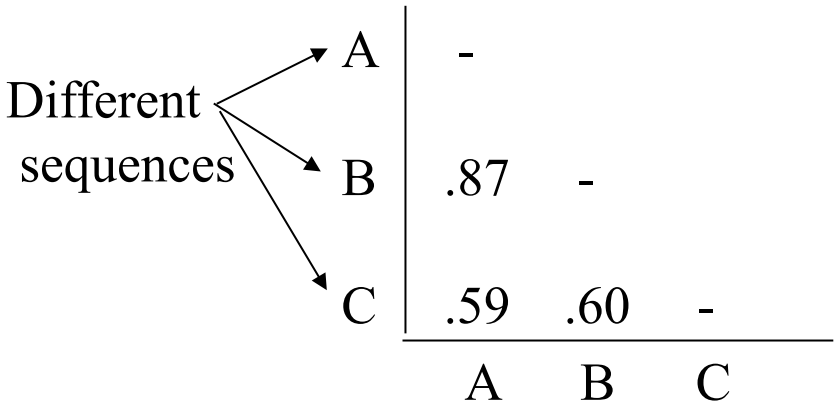


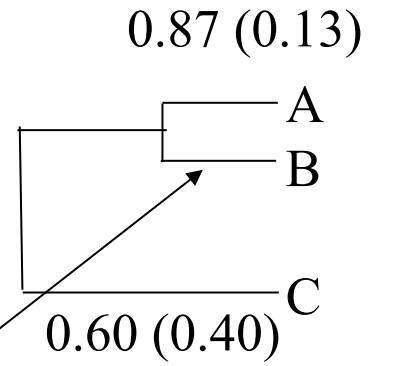
Each number represents the number of exact matches divided by the sequence length (ignoring gaps). Thus, the higher the number the more closely related the two sequences are.

In this similarity (distance) matrix sequence A is 87% identical to sequence B

Step 2-Create Guide Tree

Use the Similarity Matrix to create a Guide Tree to determine the "order" of the sequences.

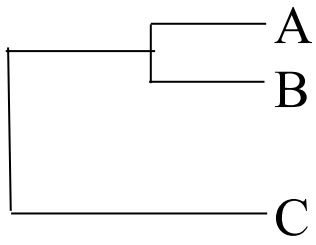




Guide Tree

Branch length proportional to estimated divergence between A and B (0.13)

Step 3-Progressive Alignment



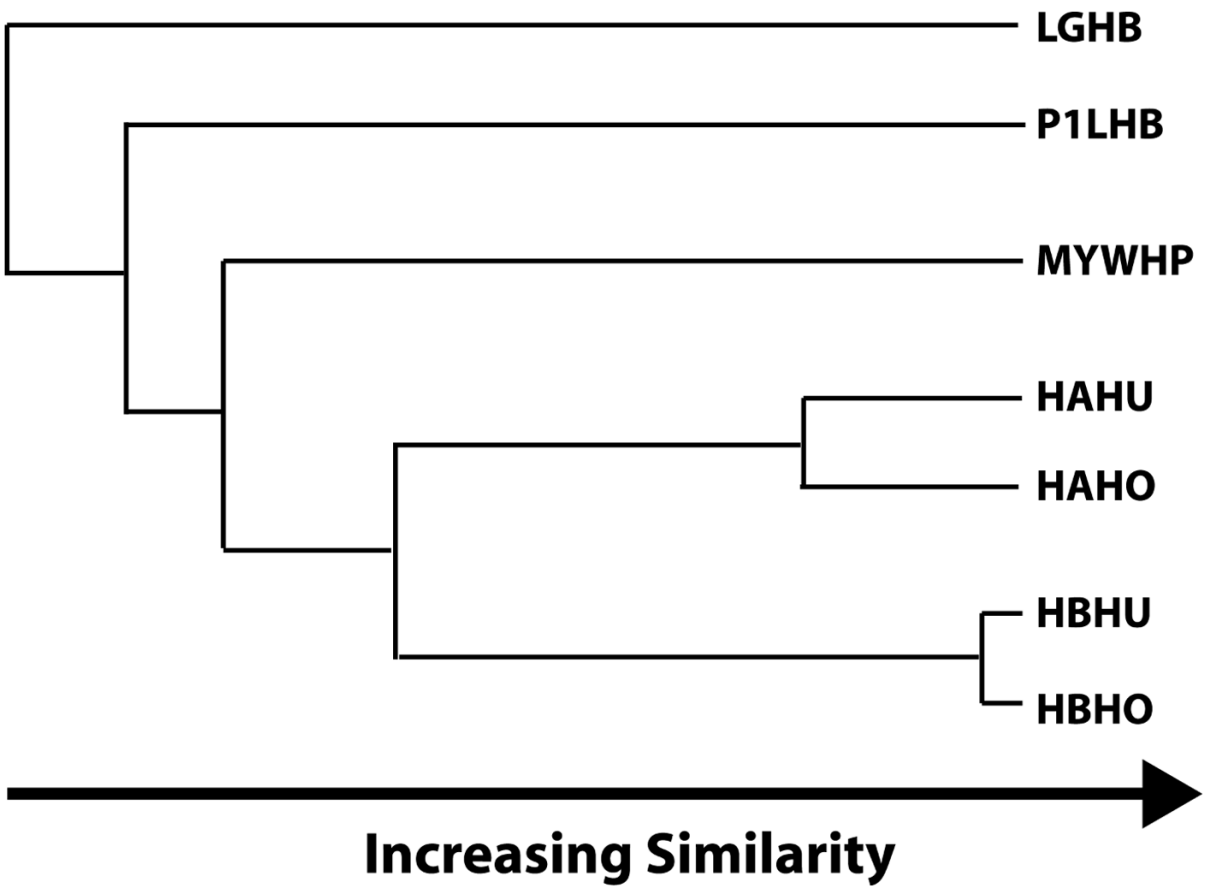
Guide Tree

Align A and B first. Then add sequence C to the previous alignment. In the closely aligned sequences and gaps are given a heavier weight than more divergent sequences.

How does it work?

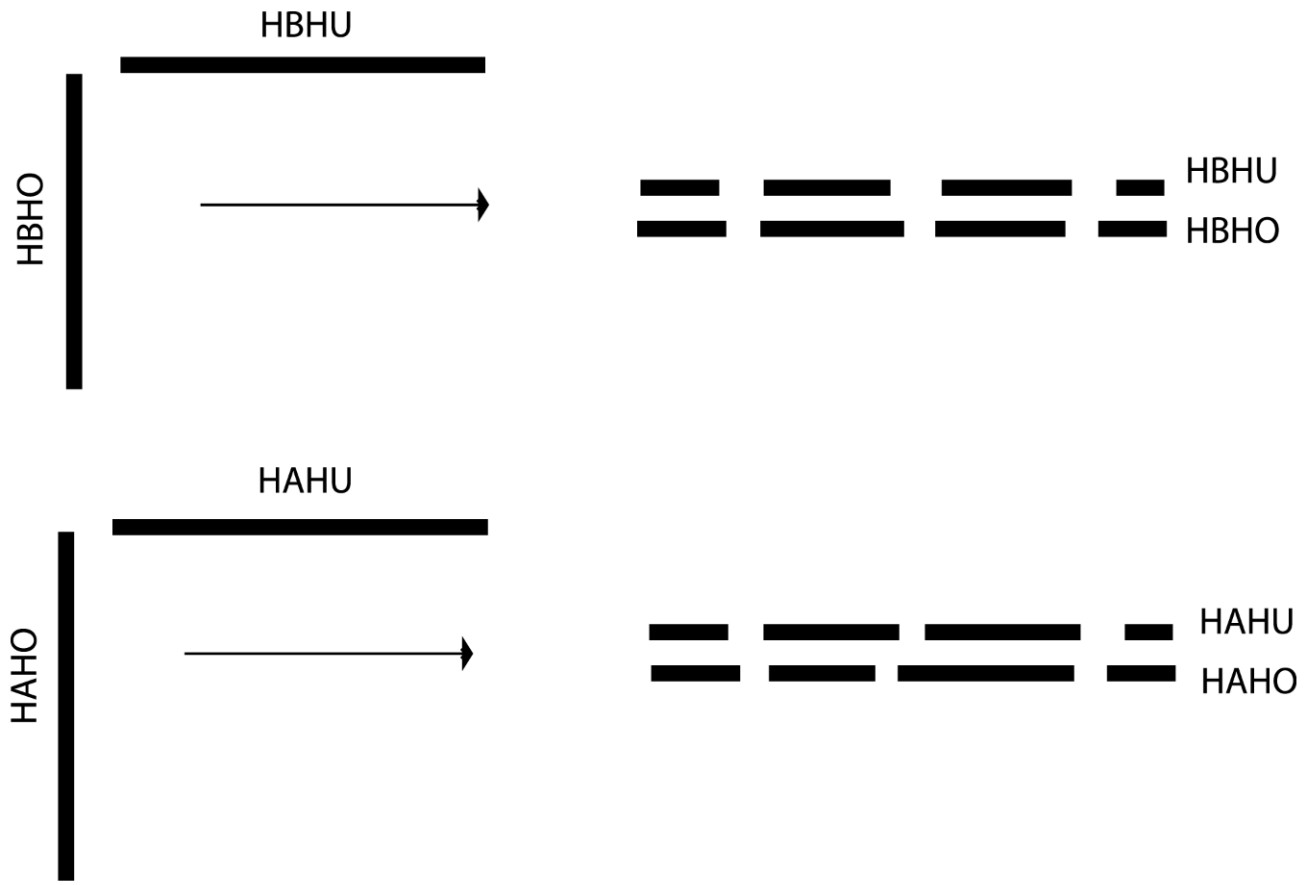
- Starting with a group of 7 sequences from different species
- Do pairwise alignments between all 7 sequences
- Score given for similarity, higher score indicates more similar

	HAHU	HBHU	HAHO	HBHO	MYWHP	P1LHB	LGHB
HAHU							
HBHU	21.1						
HAHO	32.9	19.7					
HBHO	20.7	**39.0**	20.4				
MYWHP	11.0	9.8	10.3	9.7			
P1LHB	9.3	8.6	9.6	8.4	7.0		
LGHB	7.1	7.3	7.5	7.4	7.3	4.3	

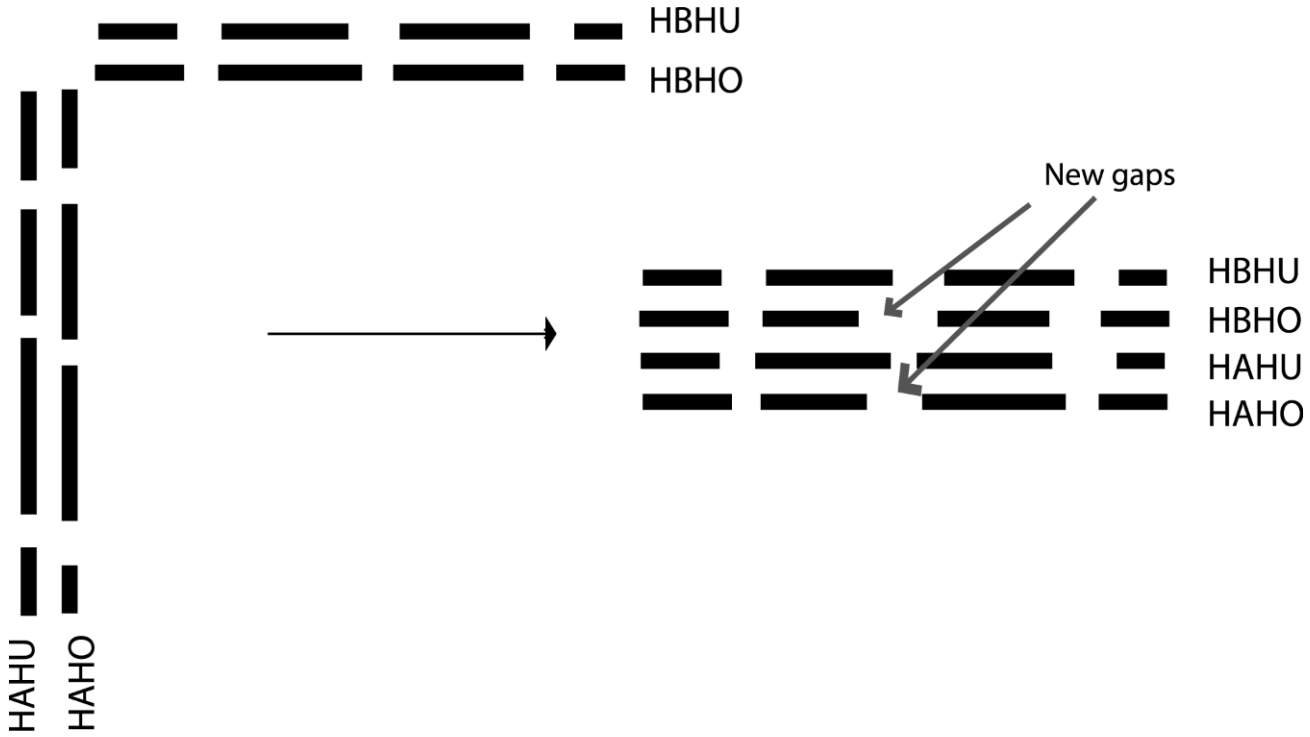


- Cluster the sequences by similarity to create a guide tree
- Branch length is proportional to estimated divergence between the two sequences

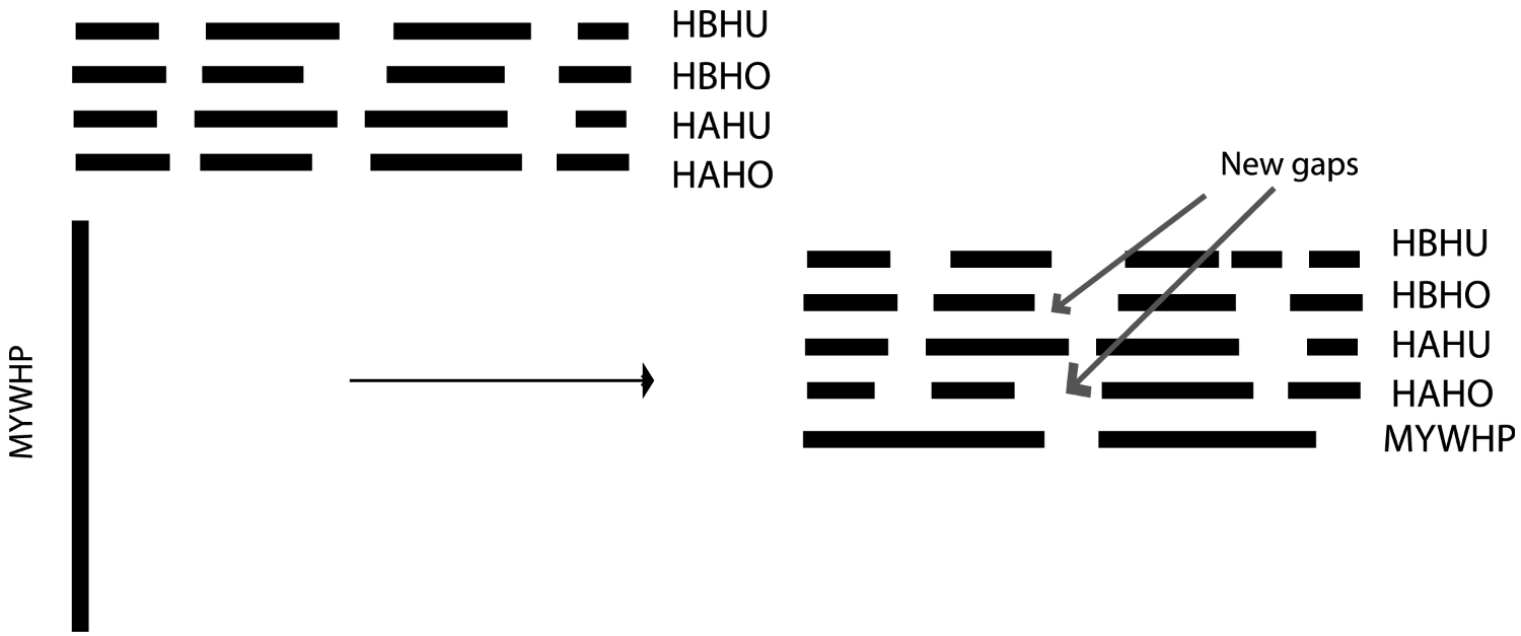
Progressive Alignment



Progressive Alignment



Progressive Alignment



BWT of ‘tatatatata$’

t	a	t	a	t	a	t	a	t	a	$
a	t	a	t	a	t	a	t	a	$	t
t	a	t	a	t	a	t	a	$	t	a
a	t	a	t	a	t	a	$	t	a	t
t	a	t	a	t	a	$	t	a	t	a
a	t	a	t	a	$	t	a	t	a	t
t	a	t	a	$	t	a	t	a	t	a
a	t	a	$	t	a	t	a	t	a	t
t	a	$	t	a	t	a	t	a	t	a
a	$	t	a	t	a	t	a	t	a	t
$	t	a	t	a	t	a	t	a	t	a

A

$	t	a	t	a	t	a	t	a	t	a
a	$	t	a	t	a	t	a	t	a	t
a	t	a	$	t	a	t	a	t	a	t
a	t	a	t	a	$	t	a	t	a	t
a	t	a	t	a	t	a	$	t	a	t
a	t	a	t	a	t	a	t	a	$	t
t	a	$	t	a	t	a	t	a	t	a
t	a	t	a	$	t	a	t	a	t	a
t	a	t	a	t	a	$	t	a	t	a
t	a	t	a	t	a	t	a	$	t	a
t	a	t	a	t	a	t	a	t	a	$

B

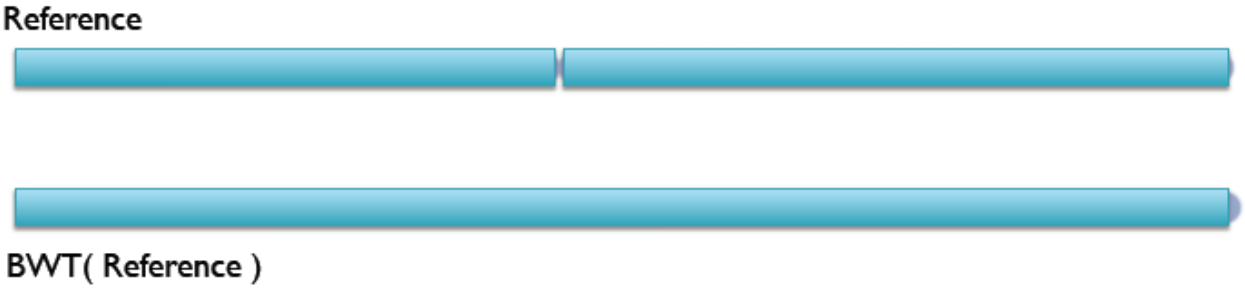
In A the word ‘tatatatata$’ undergoes cyclic shift andit is sorted in B
Note that the BWT(tatatatata$) is a word, pointed by the arrow (atttttaaaa$) with good clustering of T’s and A’s so that it can be written in a more compact way. The DNA is from an alphabet of 4 symbols (a,c,g,t) so this clustering happens very often.

More Burrows-Wheeler

Input	SIX.MIXED.PIXIES.SIFT.SIXTY.PIXIE.DUST.BOXES
Burrows-Wheeler Output	TEXYDST.E.IXIXIXXSSMPPS.B..E.S.EUSFXDIIOIIIT

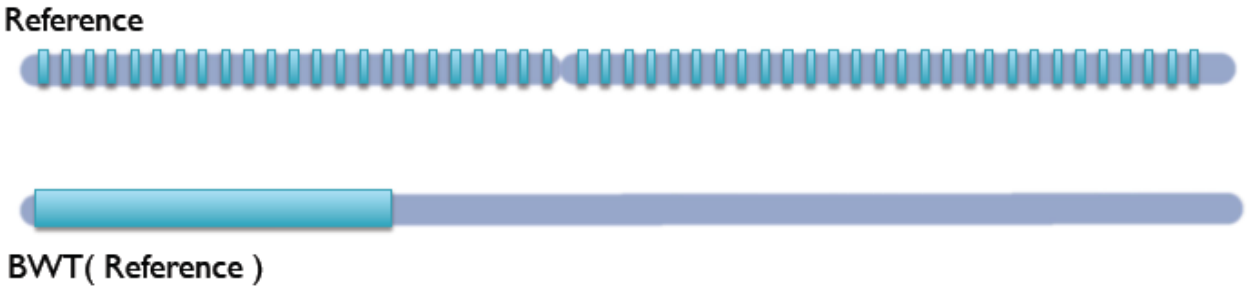
Repeated characters mean that it is easier to compress

More Burrows-Wheeler



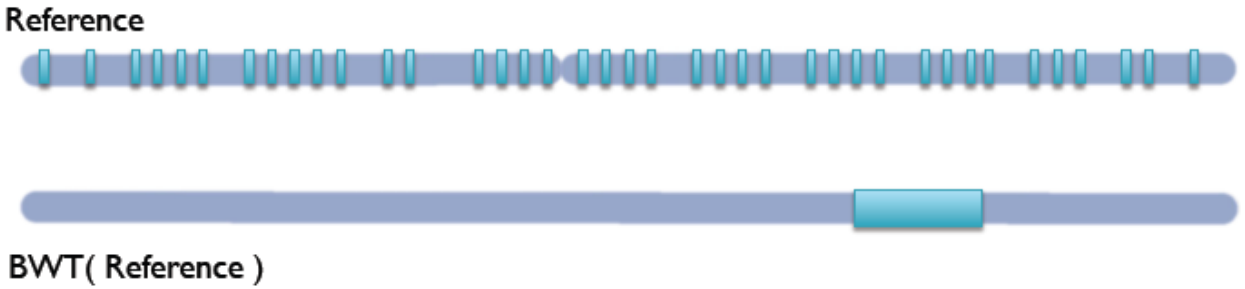
Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



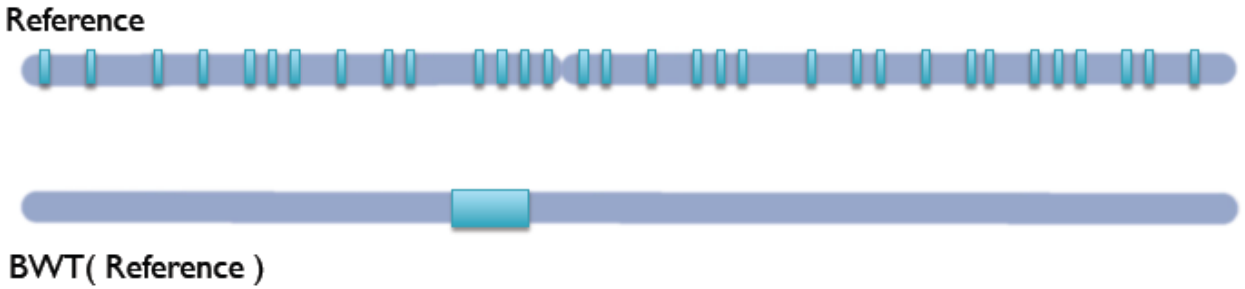
Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



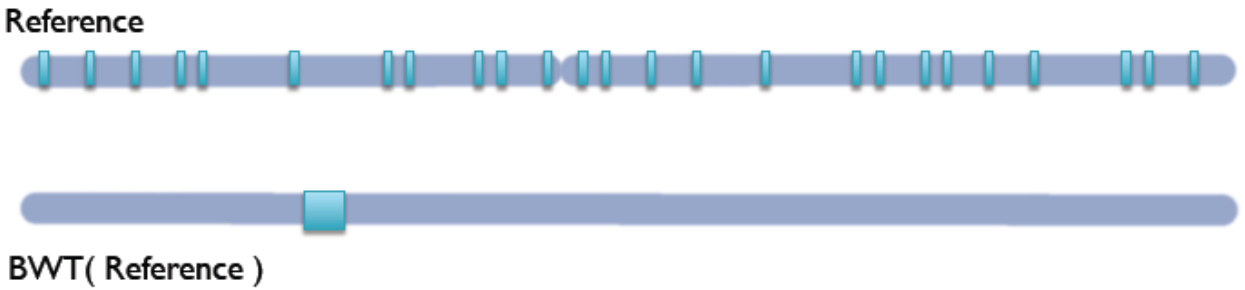
Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



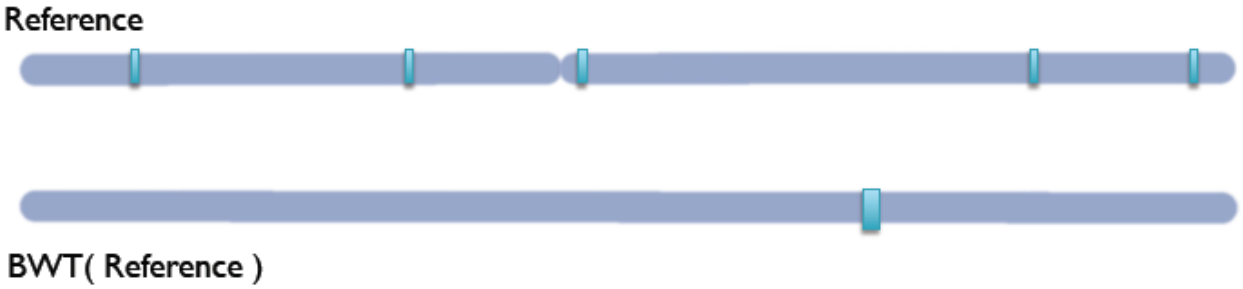
Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



Query:
AATGATACGGCGACCACCGAGATCTA

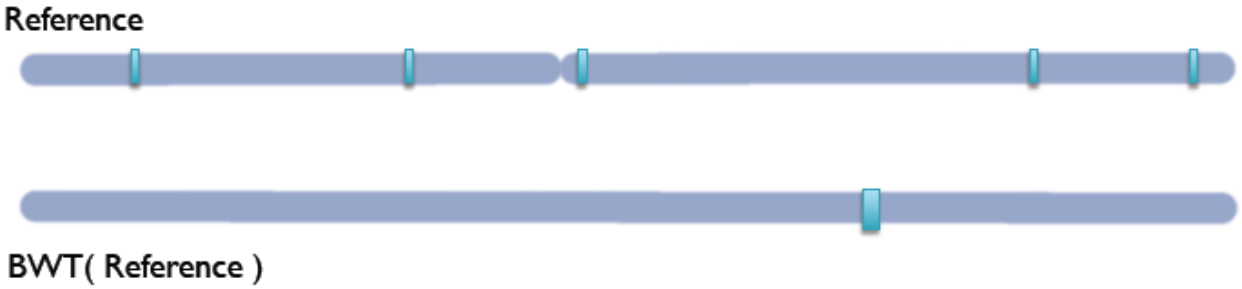
More Burrows-Wheeler

Reference

BWT(Reference)

Query:
AATGATACGGCGACCACCGAGATCTA

More Burrows-Wheeler



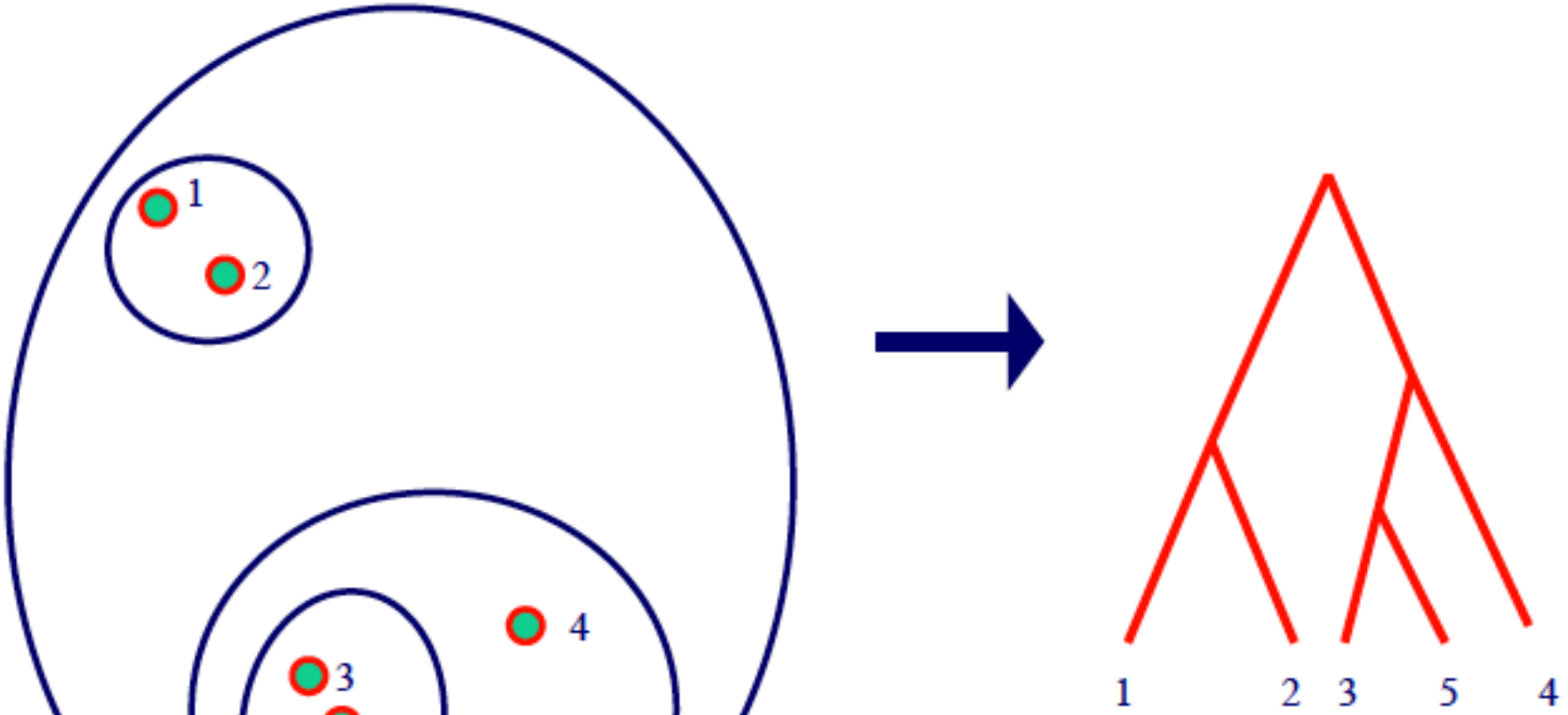
Query:
AATGTTACGGCGACCACCGAGATCTA

More Burrows-Wheeler

Query:

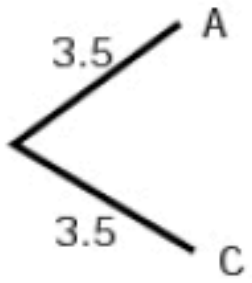
AATGTTACGGCGACCACCGAGATCTA

UPGMA



UPGMA

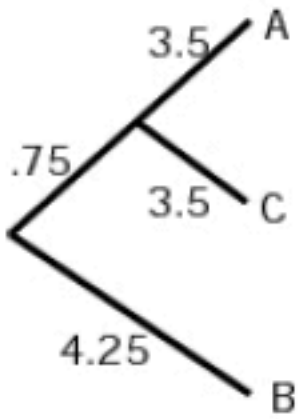
	A	B	C	D
A	0			
B	8	0		
C	7	9	0	
D	12	14	11	0



$$M_{B(AC)} = (M_{BA} + M_{BC})/2 = (8+9)/2=8.5$$

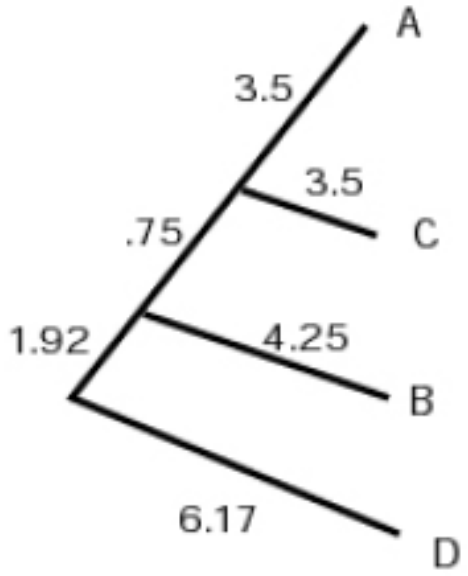
$$M_{D(AC)} = (M_{DA} + M_{DC})/2= (12+11)/2=11.5$$

	AC	B	D
AC	0		
B	8.5	0	
D	11.5	14	0



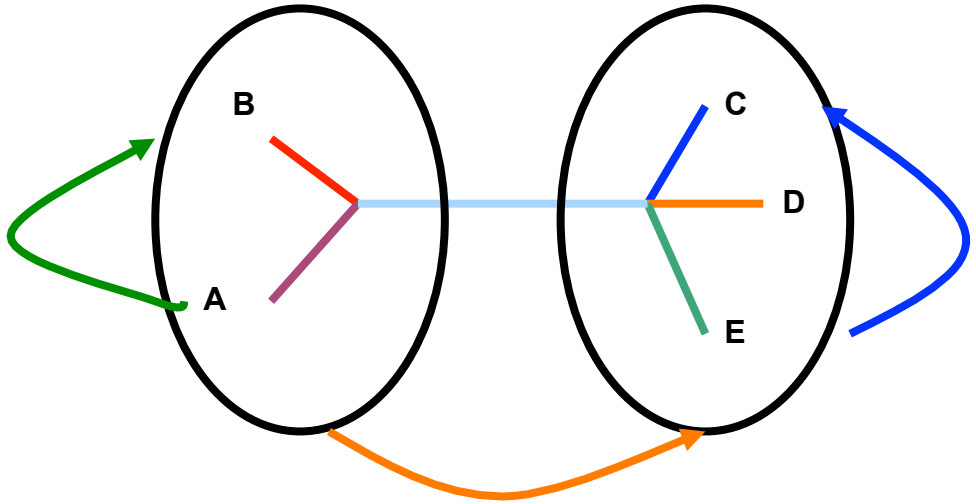
$$M_{(ABC)D} = (M_{AD} + M_{BD} + M_{CD})/3 = (12+14+11)/3$$

	ABC	D
ABC	0	
D	12.33	0



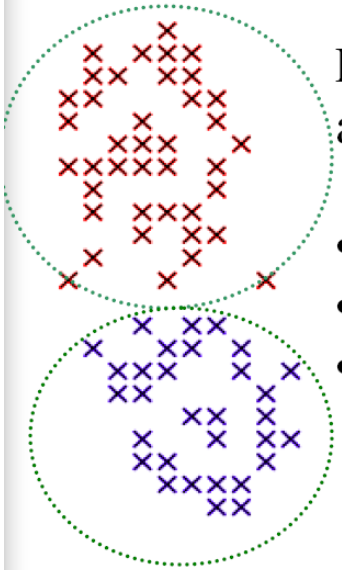
Neighbor Joining

- If A and B are joined:



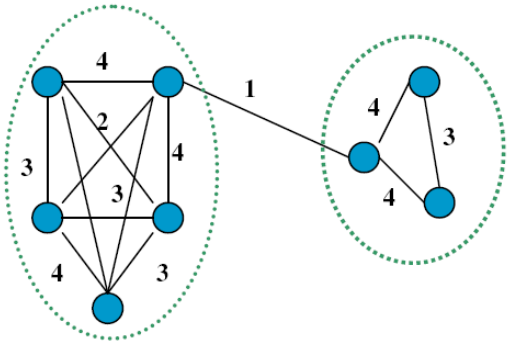
$$S_{mn} = \frac{\sum \left(d_{im} + d_{in} \right)}{2(N-2)} + \frac{d_{mn}}{2} + \frac{\sum d_{ij}}{N-2}$$

Vector clustering (K means) Graph clustering (Markov Clustering algorithm)

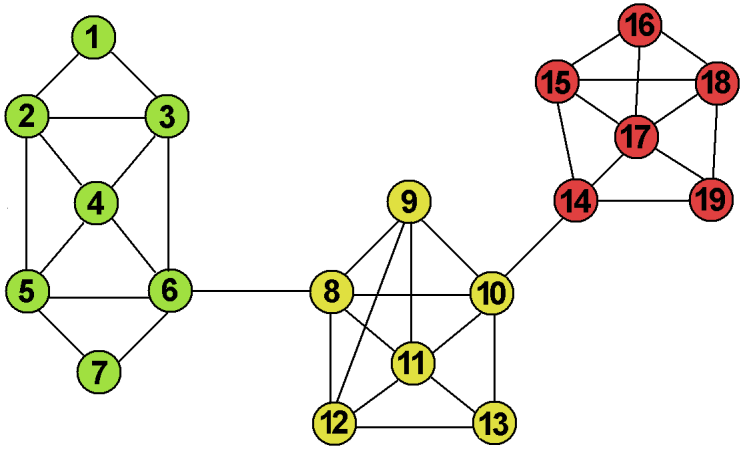


Each point has a vector, i.e.

- x coordinate
- y coordinate
- color



Each vertex is connected to others by (weighted or unweighted) edges.



0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	1	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	1	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0

MCL animation

- Found at

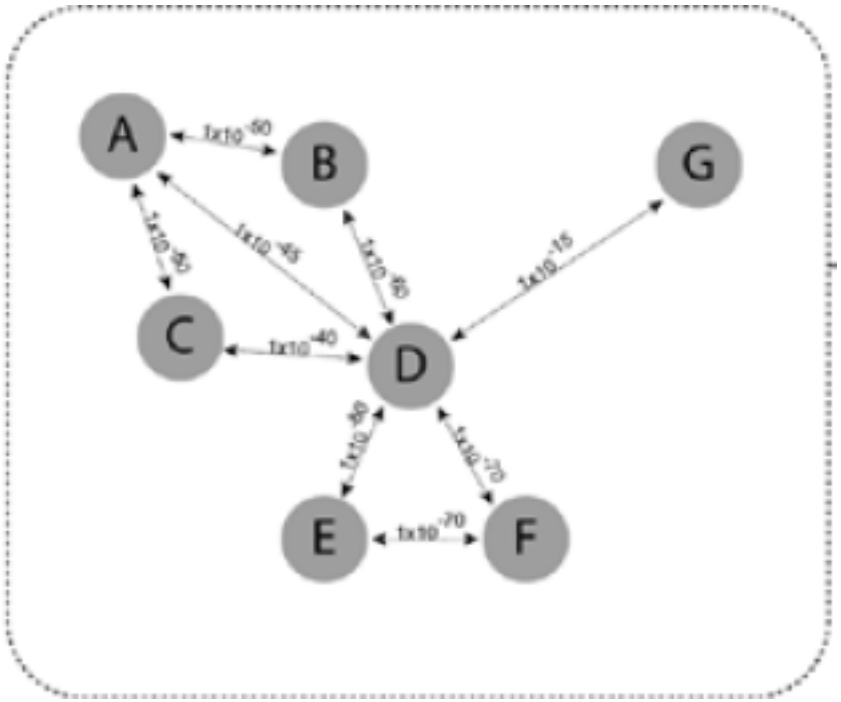
www.micans.org/mcl/ani/mcl-animation.html

© 2002 Oxford University Press
Nucleic Acids Research, 2002, Vol. 30, No. 7 1575–1584

An efficient algorithm for large-scale detection of protein families

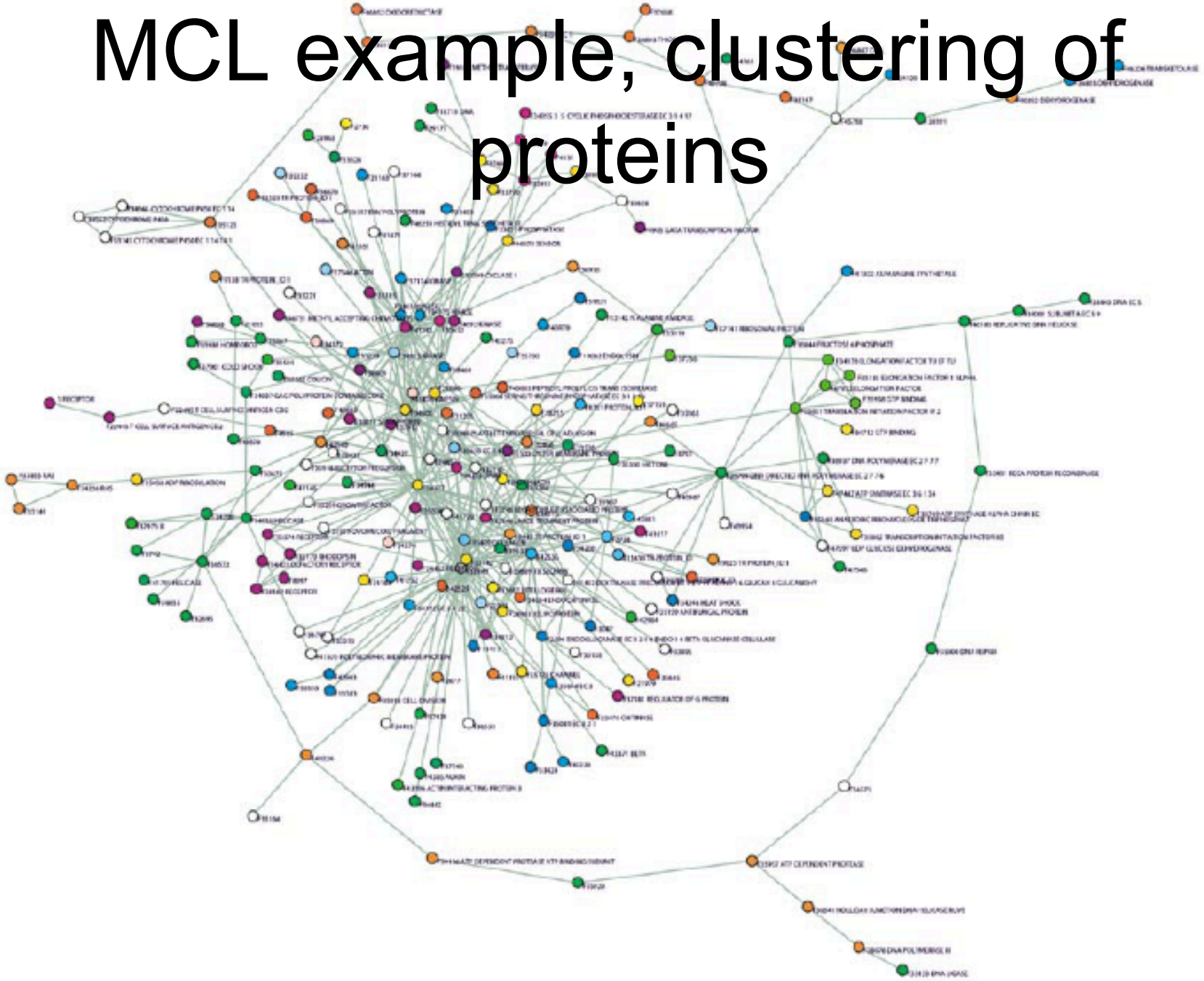
A. J. Enright*, S. Van Dongen[1] and C. A. Ouzounis

Computational Genomics Group, The European Bioinformatics Institute, EMBL Cambridge Outstation, Cambridge CB10 1SD, UK and [1]Centrum voor Wiskunde en Informatica, Kruislaan 413, NL-1098 SJ Amsterdam, The Netherlands

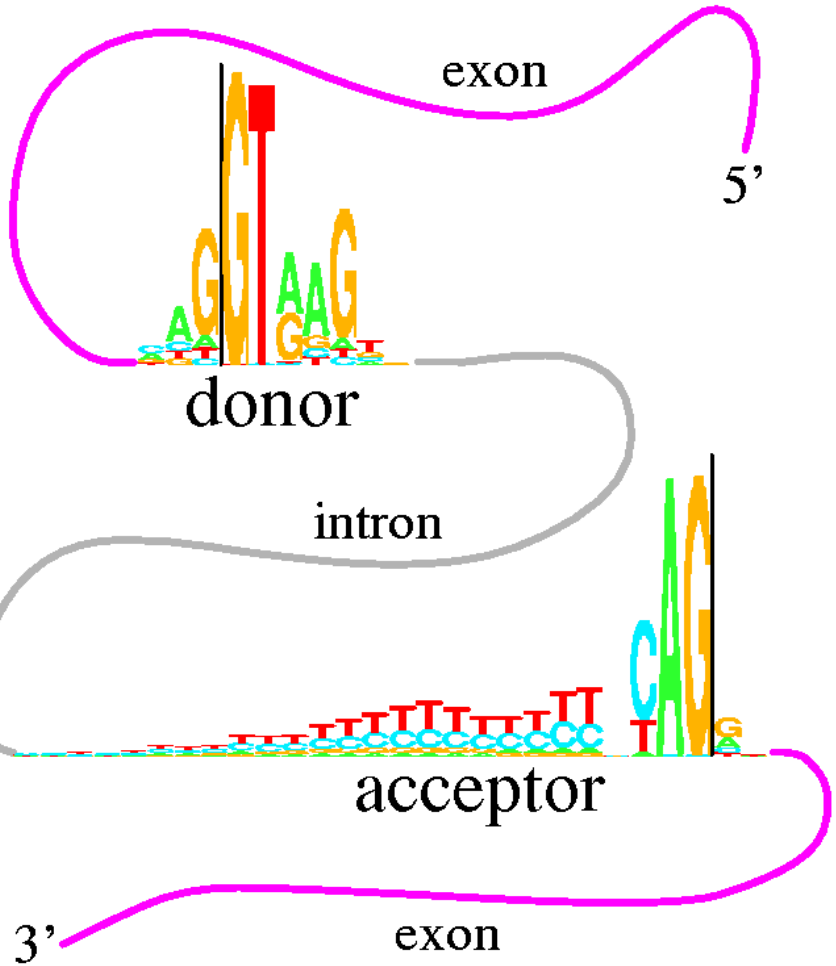


	A	B	C	D	E	F	G
A	0.42	0.24	0.20	0.11	0.00	0.00	0.00
B	0.20	0.48	0.24	0.15	0.00	0.00	0.00
C	0.20	0.00	0.40	0.10	0.00	0.00	0.00
D	0.18	0.28	0.16	0.24	0.32	0.29	0.13
E	0.00	0.00	0.00	0.19	0.40	0.29	0.00
F	0.00	0.00	0.00	0.17	0.28	0.42	0.00
G	0.00	0.00	0.00	0.04	0.00	0.00	0.87

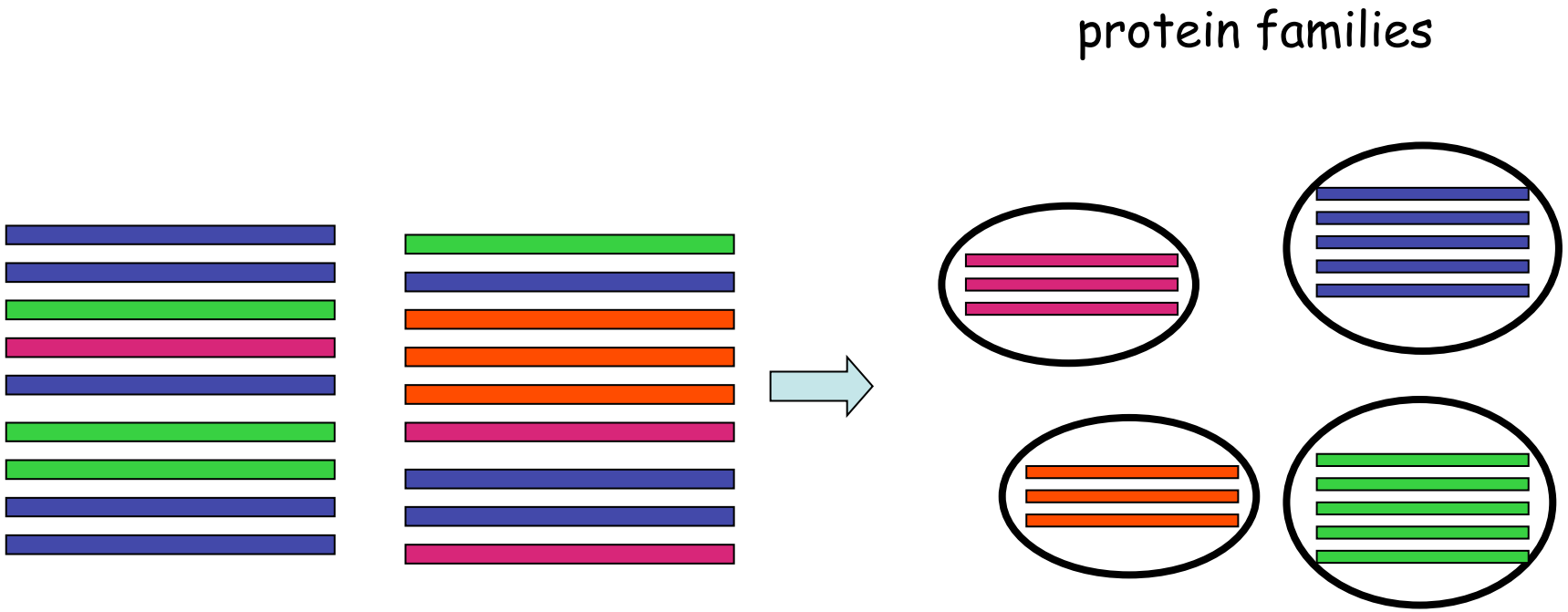
MCL example, clustering of proteins



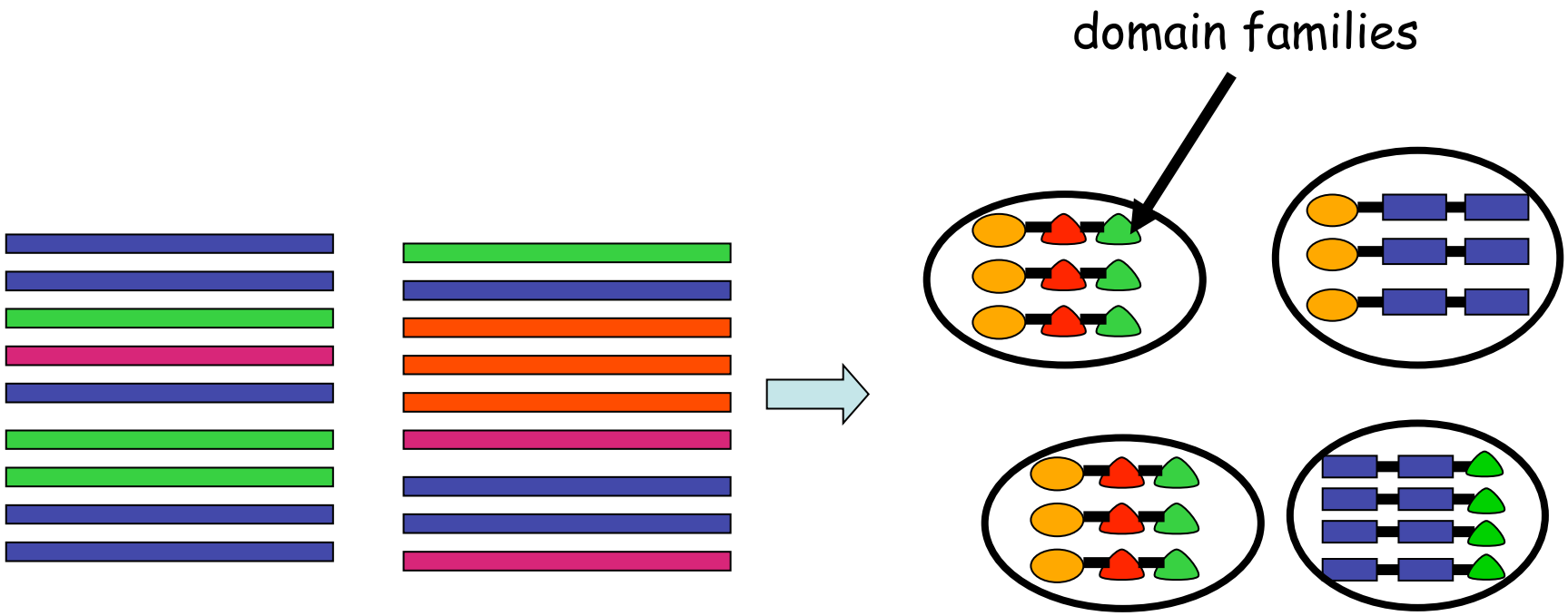
Applications of HMM



Applications of HMM: Identify protein families in the genomes



Applications of HMM: Identify domains and protein families in the genomes



The Probability that they are all Fair die rolls

Let the sequence of rolls be:

x = 1, 2, 1, 5, 6, 2, 1, 5, 2, 4

Then, what is the likelihood of

π = F, F, …, F?

(say initial probs $a_{0Fair} = ½$, $a_{oLoaded} = ½$)

$½ \times P(1 \mid Fair)\ P(Fair \mid Fair)\ P(2 \mid Fair)\ P(Fair \mid Fair) \ldots P(4 \mid Fair) =$

$½ \times (1/6)^{10} \times (0.95)^{9} = .000000005211586472 11 \sim= 0.5 \times 10^{-9}$

The ½ comes from the initial probability. Each value has an emission probability of (1/6). There 9 transitions from Fair to Fair each with a probability of .95.

Sequence of rolls have many 6's

Let the sequence of rolls be:

x = 1, 6, 6, 5, 6, 2, 6, 6, 3, 6

Now, what is the likelihood π = F, F, …, F?

$\frac{1}{2} \times (1/6)^{10} \times (0.95)^{9} = 0.5 \times 10^{-9}$, same as before

What is the likelihood

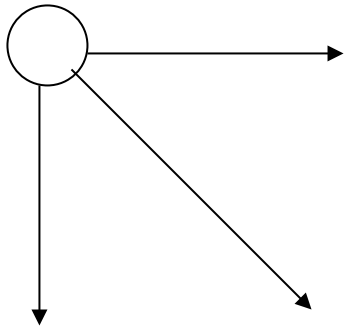
π = L, L, …, L?

$\frac{1}{2} \times (1/10)^{4} \times (1/2)^{6} (0.95)^{9} = .00000049238235134735 \sim= 0.5 \times 10^{-7}$

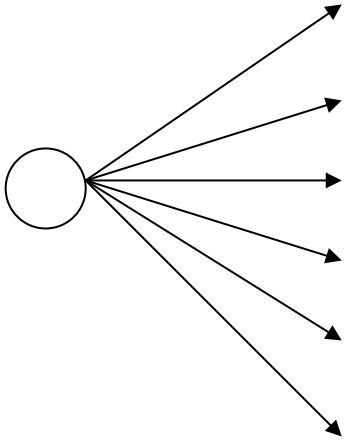
There are 4 non-6 values with a emission probability of 1/10 and 6 values of 6 each with an emission probability of (1/2) thus (1/2)^6. There are 9 state transitions of loaded die with a probability of .95 per transition.

So, it is 100 times more likely the die is loaded. However, this is not the only possible parse because some could be fair and some could be loaded.

Decoding Problem vs. Alignment Problem



Valid directions in the *alignment problem.*



Valid directions in the *decoding problem.*

Viterbi gets it right more often than not

```
Rolls     3151162464466442453211316311641521336251445436316566265666666
Die       FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLL

Rolls     6511664531326512456366646316366631623264552352666666251516 31
Die       LLLLLLFFFFFFFFFFFLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLFFFFFFFFF
Viterbi   LLLLLLFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls     2225554416665665635643243641315134651463534111264146262533 56
Die       FFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls     3661636664662325344136616611632525624622552652522664353533 36
Die       LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi   LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls     2331216253644144323351632436336655624666626326666123552452 42
Die       FFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLFFFFFFFFFF
```

Viterbi Example

Let x be a long sequence with a portion of ~ 1/6 6's,
followed by a portion of ~ ½ 6's…

x = 123456123456…12345 6626364656…1626364656

Then, it is not hard to show that optimal parse is (exercise):

FFF……..F LLL……...L

The optimal parse is a set of Fairs in the beginning and a set of Loads at the end and the optimal cutting position will be at a non-6 followed by a 6. You would prefer to go one step forward and take a non-6 with a Fair state than a 6. And similarly, you would prefer to go one step back and take a 6 with a loaded die rather than a Fair state die because of the transition probabilities.

6 characters "123456" parsed as F, contribute $.95^6 \times (1/6)^6 = 1.6 \times 10^{-5}$
parsed as L, contribute $.95^6 \times (1/2)^1 \times (1/10)^5 = 0.4 \times 10^{-5}$

"162636" parsed as F, contribute $.95^6 \times (1/6)^6 = 1.6 \times 10^{-5}$
parsed as L, contribute $.95^6 \times (1/2)^3 \times (1/10)^3 = 9.0 \times 10^{-5}$

Underflows and Logs

Underflows are a significant problem

$$P[x_1,\ldots, x_i, \pi_1, \ldots, \pi_i] = a_{0\pi 1}\, a_{\pi 1 \pi 2} \ldots\ldots a_{\pi i}\, e_{\pi 1}(x_1) \ldots\ldots e_{\pi i}(x_i)$$

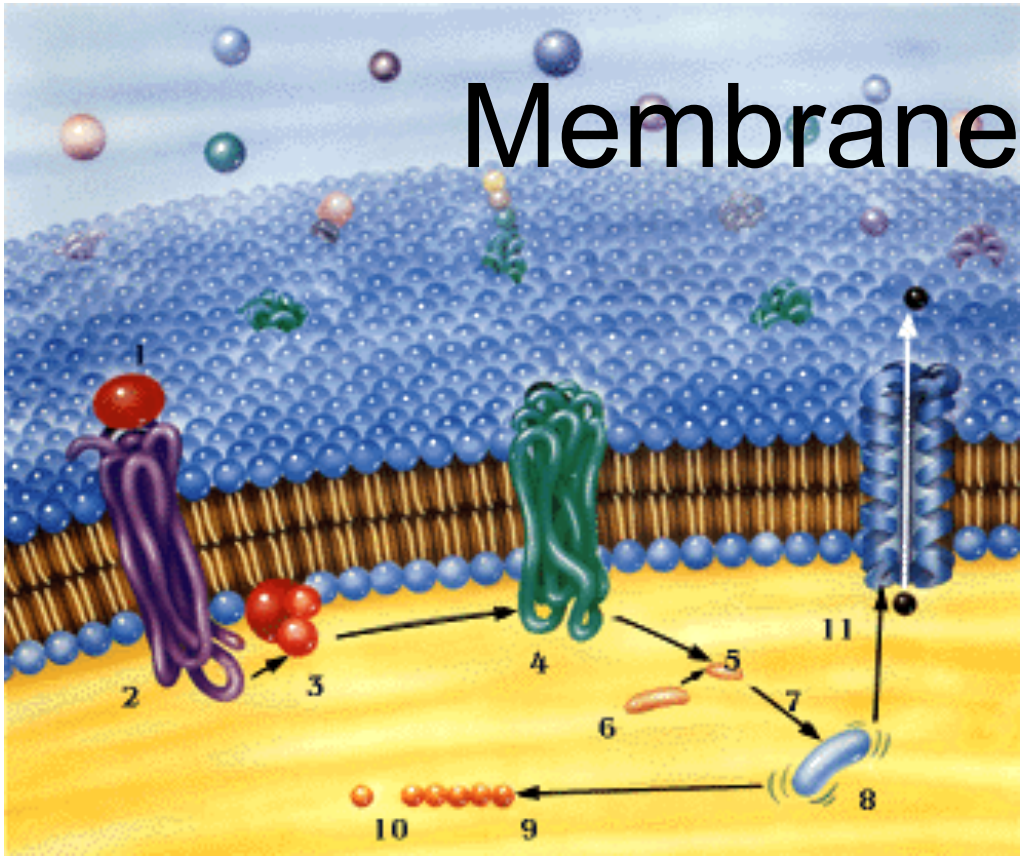
These numbers become extremely small probabilities and underflows the computer stored value.

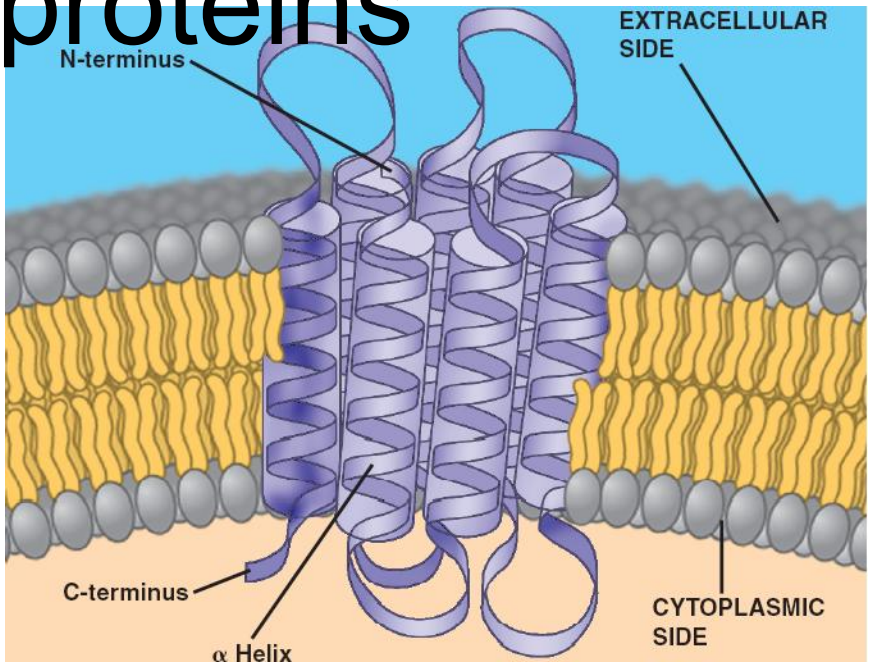
Solution: Take the logs of all values

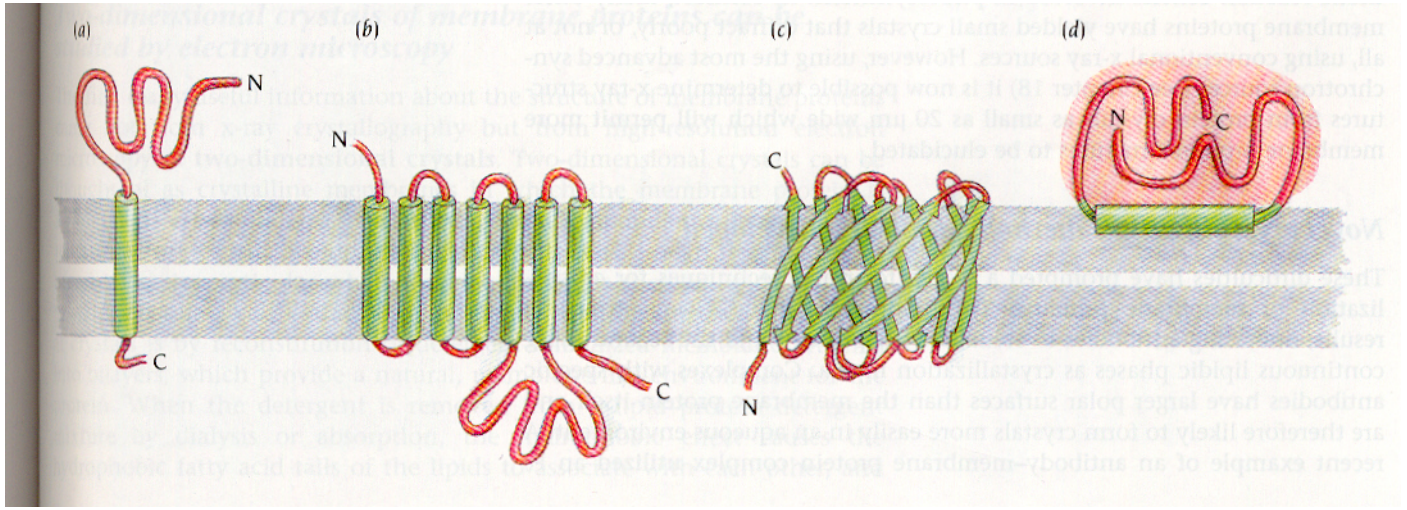
$$V_l(i) = \log e_k(x_i) + \max_k [V_k(i\text{-}1) + \log a_{kl}]$$

Should initial V0(0) to 0 instead of 1.

Membrane proteins







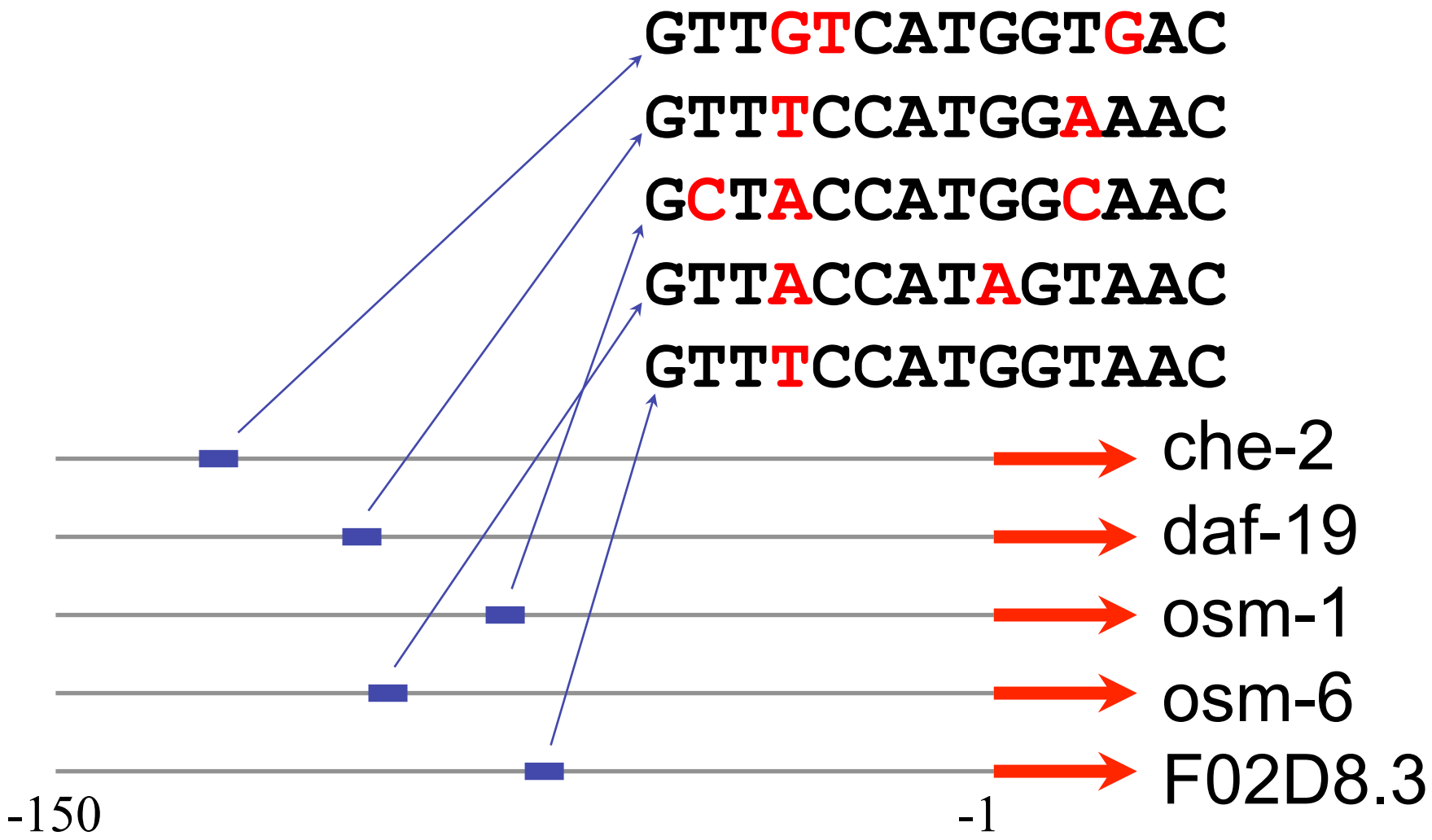
Example for TMHMM
www.cbs.dtu.dk/services/TMHMM/

>gi|218694017|ref|YP_002401684.1| membrane protein; channel [Escherichia coli 55989]
MQDLISQVEDLAGIEIDHTTSMVMIFGIIFLTAVVVHIILHWVVLRTFEKRAIASS
RLWLQIITQNKLFH
RLAFTLQGIIVNIQAVFWLQKGTEAADILTTCAQLWIMMYALLSVFSLLDVILNL
AQKFPAASQLPLKGI
FQGIKLIGAILVGILMISLLIGQSPAILISGLGAMAAVLMLVFKDPILGLVAGIQLS
ANDMLKLGDWLEM
PKYGADGAVIDIGLTTVKVRNWDNTITTIPTWSLVSDSFKNWSGMSASGGRR
IKRSISIDVTSIRFLDED
EMQRLNKAHLLKPYLTSRHQEINEWNRQQGSTESILNLRRMTNIGTFRAYLN
EYLRNHPRIRKDMTLMVR
QLAPGDNGLPLEIYAFTNTVVWLEYESIQADIFDHIFAIVEEFGLRLHQSPTGN
DIRSLAGAFKQ

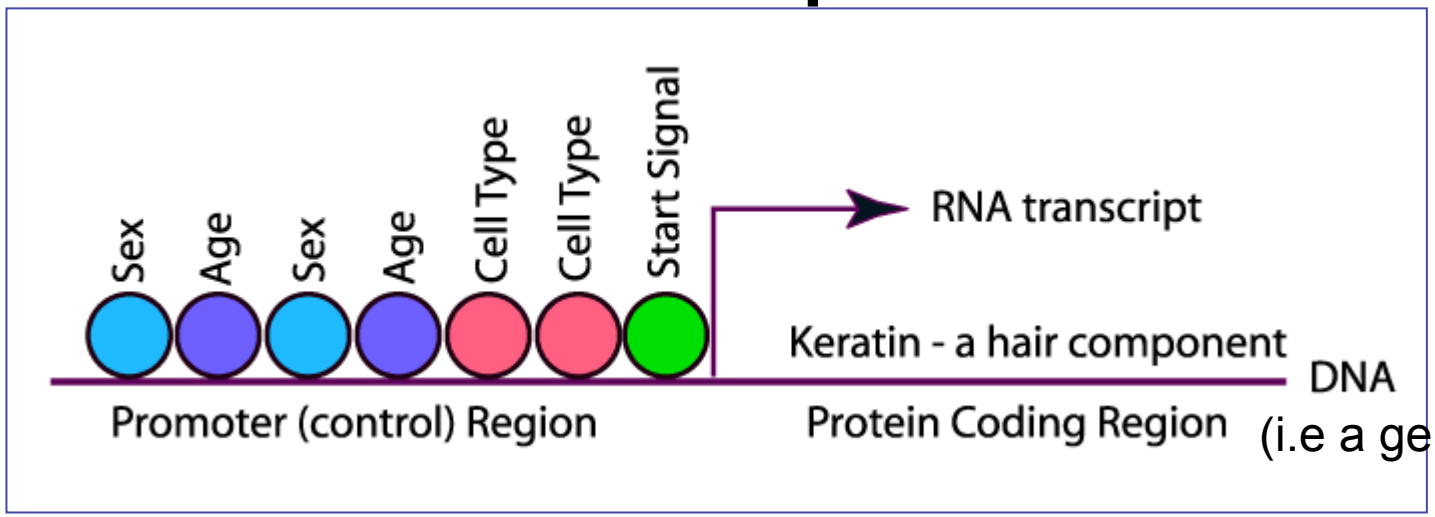
Gibbs Sampling server

http://bayesweb.wadsworth.org/cgi-bin/gibbs.8.pl?data_type=DNA

Examples of regulatory sites

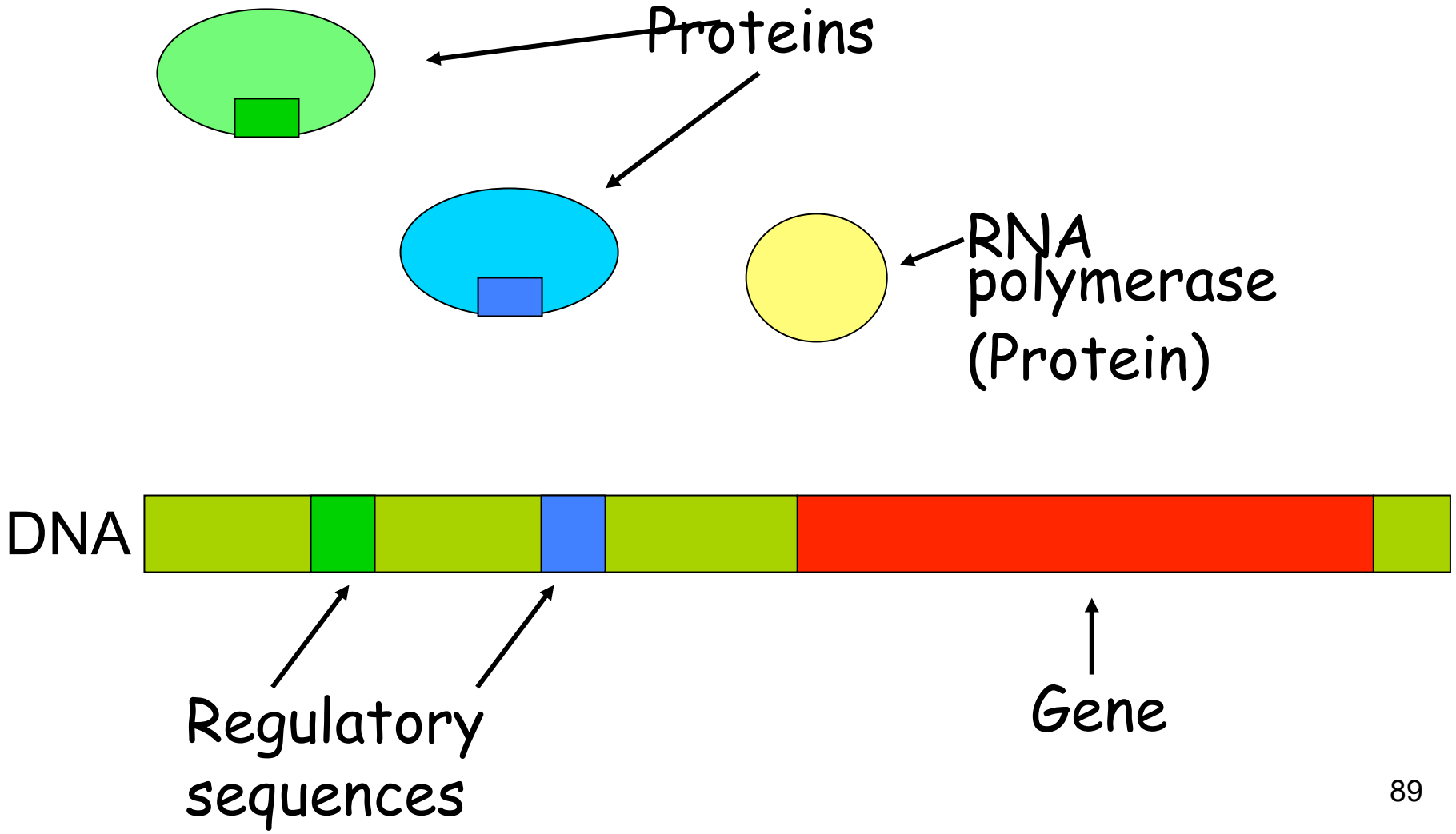


The Cell is a Computer in Soup

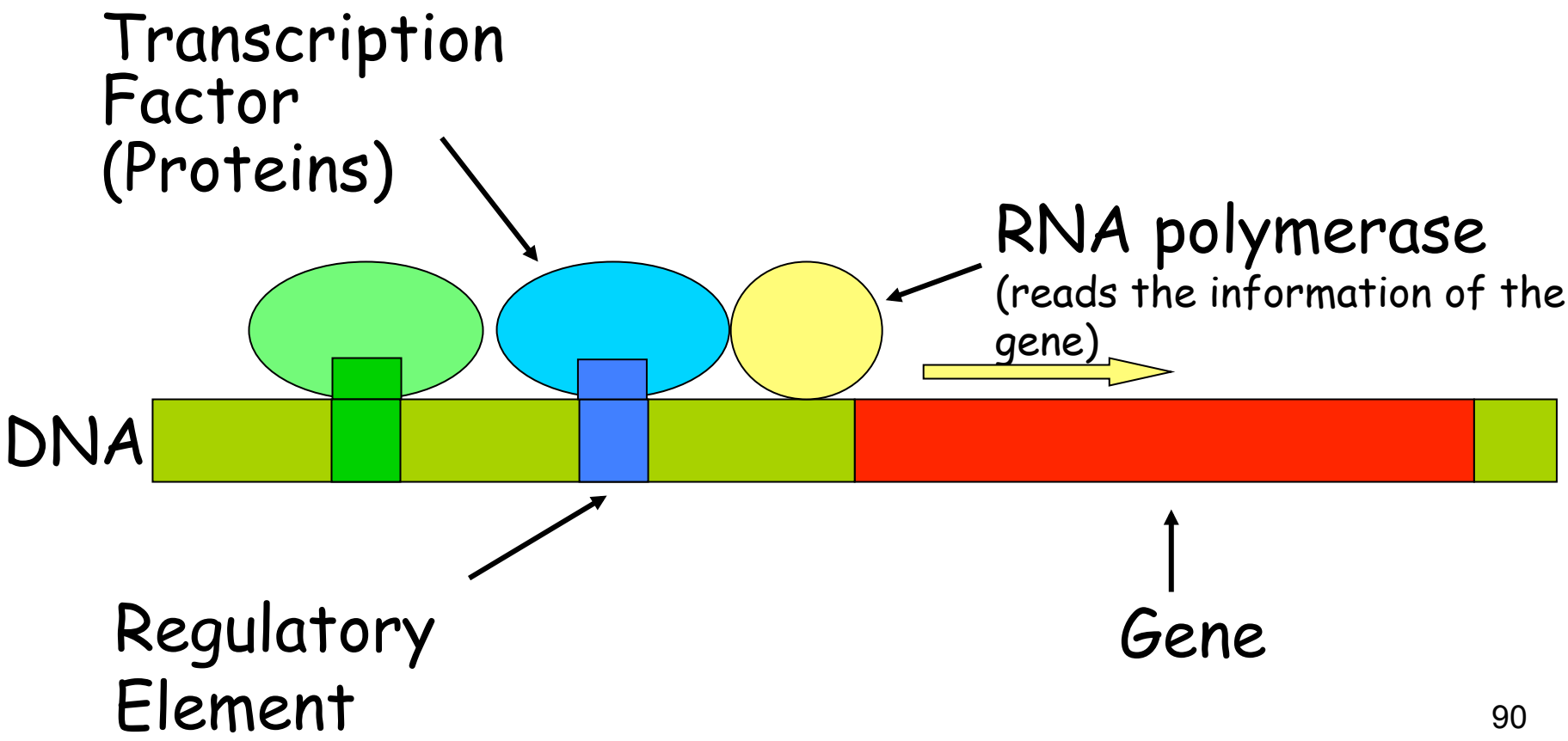


Idealized promoter for a gene involved in making hair. Proteins that bind to specific DNA sequences in the promoter region together turn a gene on or off. These proteins are themselves regulated by their own promoters leading to a gene regulatory network with many of the same properties as a neural network.

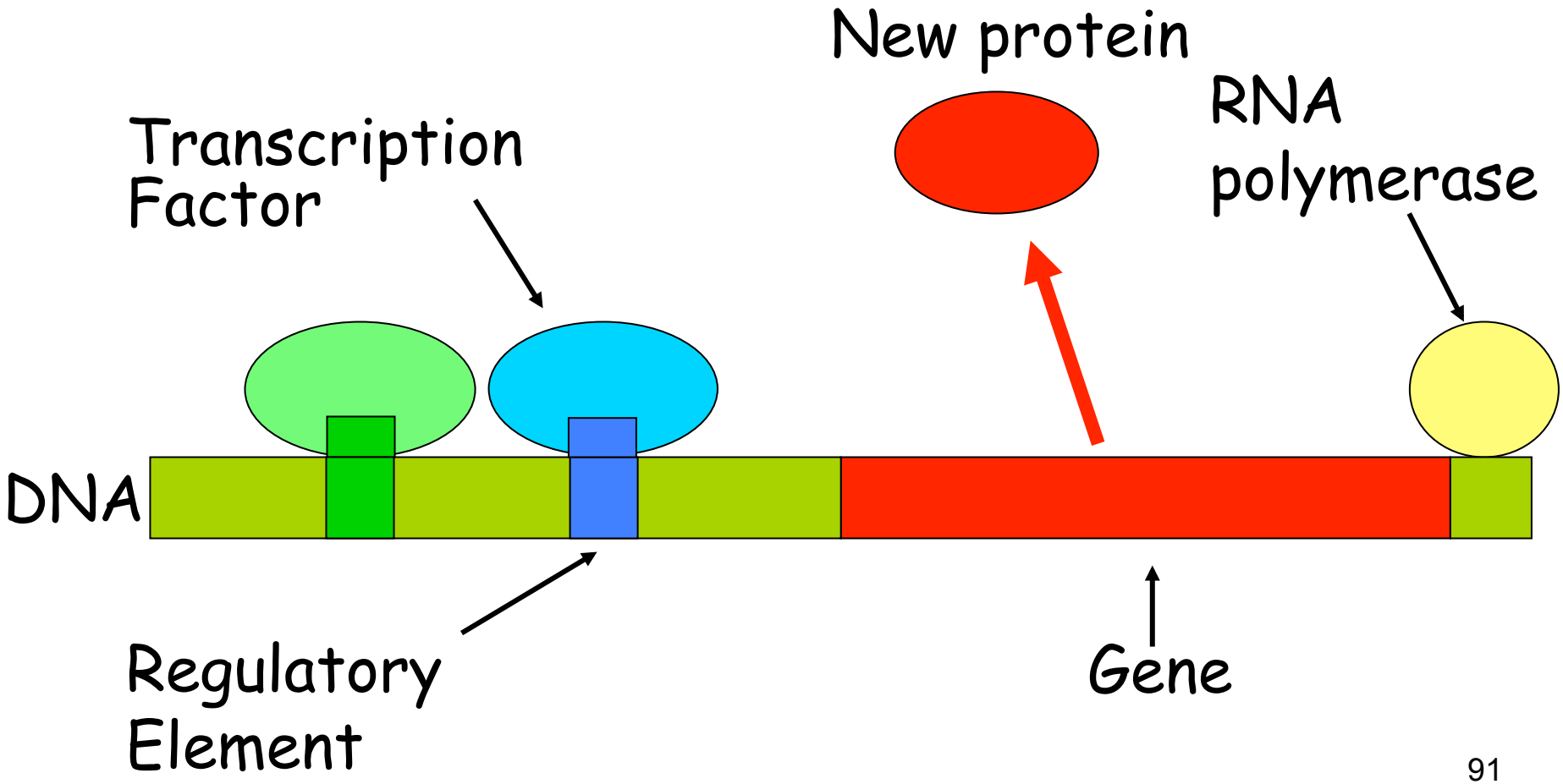
Regulation of Genes



Regulation of Genes



Regulation of Genes



Random Sample

atgaccgggatactgataccgtatttggcctaggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg

acccctatttttgagcagatttagtgacctggaaaaaaaatttgagtacaaaacttttccgaatactgggcataaggtaca

tgagtatccctgggatgacttttgggaacactatagtgctctcccgatttttgaatatgtaggatcattcgccagggtccga

gctgagaattggatgaccttgtaagtgttttccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga

tccctttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaatggcccacttagtccacttatag

gtcaatcatgttcttgtgaatggatttttaactgagggcatagaccgcttggcgcacccaaattcagtgtgggcgagcgcaa

cggttttggcccttgttagaggcccccgtactgatggaaactttcaattatgagagagctaatctatcgcgtgcgtgttcat

aacttgagttggtttcgaaaatgctctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta

ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcatttcaacgtatgccgaaccgaaagggaag

ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagcttctgggtactgatagca

Implanting Motif **AAAAAAAGGGGGGG**

Where is the Implanted Motif?

atgaccgggatactgataaaaaaaaggggggggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg

acccctatttttgagcagatttagtgacctggaaaaaaaatttgagtacaaaactttccgaataaaaaaaaggggggga

tgagtatccctgggatgacttaaaaaaaaggggggtgctctcccgatttttgaatatgtaggatcattcgccagggtccga

gctgagaattggatgaaaaaaaaggggggtccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga

tccctttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaataaaaaaaaggggggcttatag

gtcaatcatgttcttgtgaatggatttaaaaaaaaggggggggaccgcttggcgcacccaaattcagtgtgggcgagcgcaa

cggttttggcccttgttagaggcccccgtaaaaaaaaggggggcaattatgagagagctaatctatcgcgtgcgtgttcat

aacttgagttaaaaaaaaggggggctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta

ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcataaaaaaaaggggggggaccgaaagggaag

ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagcttaaaaaaaaggggggga

Implanting Motif **AAAAAAAGGGGGGG** with Four Mutations

Where is the Motif???

atgaccgggatactgatagaagaaaggttgggggcgtacacattagataaacgtatgaagtacgttagactcggcgccgccg

acccctatttttgagcagatttagtgacctggaaaaaaaatttgagtacaaaactttccgaatacaataaaacggcggga

tgagtatccctgggatgacttaaaataatggagtggtgctctcccgatttttgaatatgtaggatcattcgccagggtccga

gctgagaattggatgcaaaaaaagggattgtccacgcaatcgcgaaccaacgcggacccaaaggcaagaccgataaaggaga

tccctttgcggtaatgtgccgggaggctggttacgtagggaagccctaacggacttaatataataaaggaagggcttatag

gtcaatcatgttcttgtgaatggatttaacaataagggctgggaccgcttggcgcacccaaattcagtgtgggcgagcgcaa

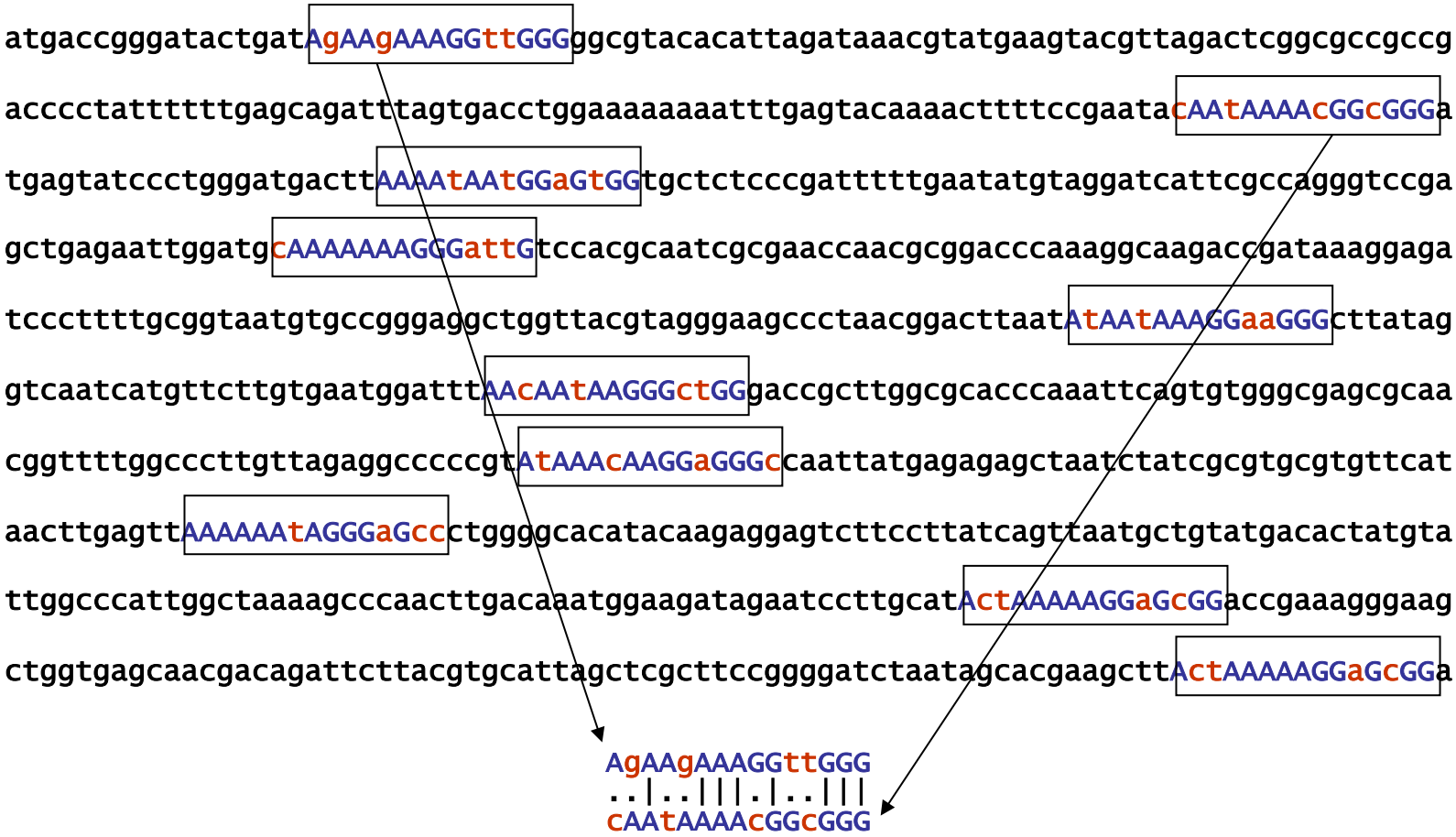
cggttttggcccttgttagaggcccccgtataaacaaggagggccaattatgagagagctaatctatcgcgtgcgtgttcat

aacttgagttaaaaaatagggagccctggggcacatacaagaggagtcttccttatcagttaatgctgtatgacactatgta

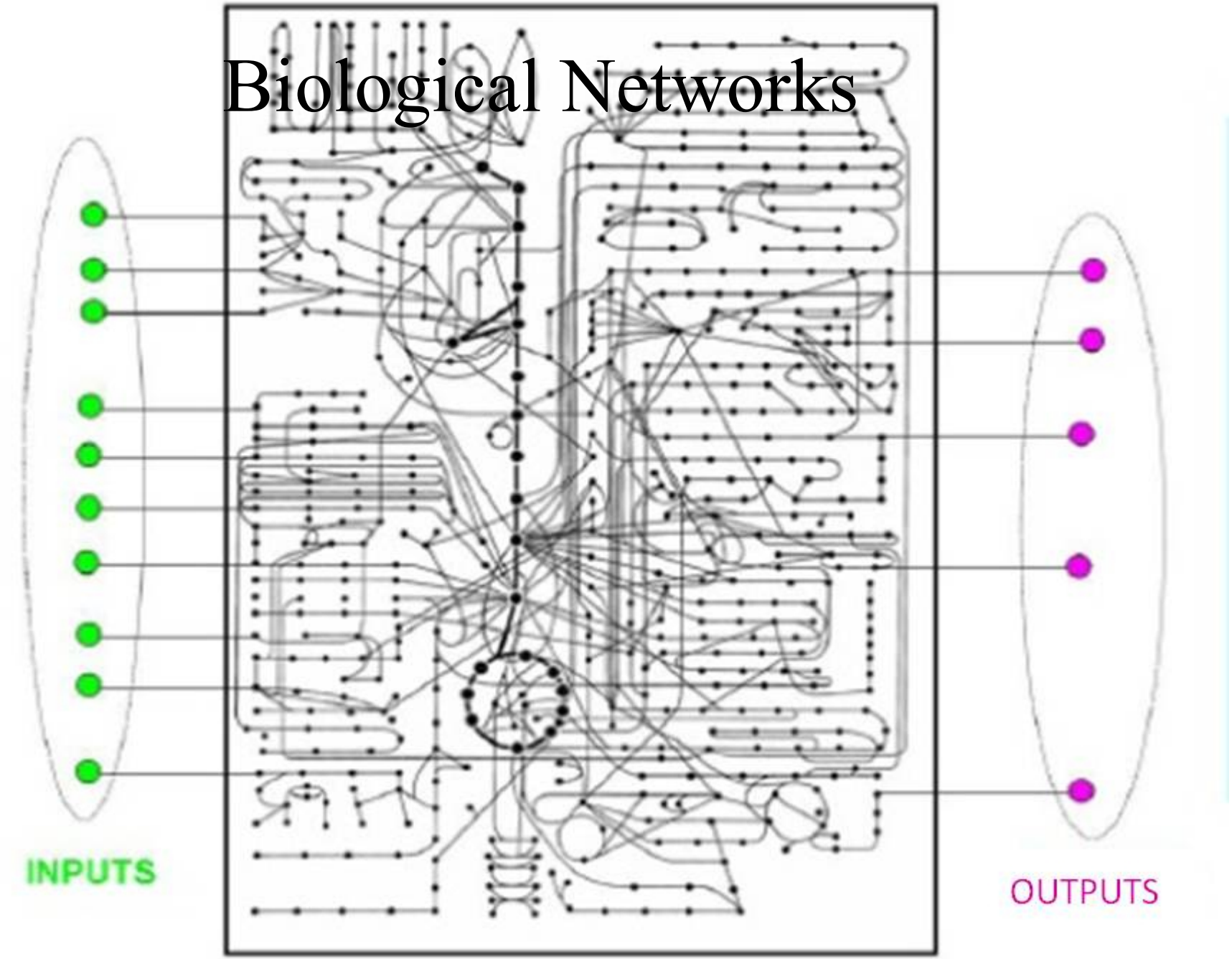
ttggcccattggctaaaagcccaacttgacaaatggaagatagaatccttgcatactaaaaaggagcggaccgaaagggaag

ctggtgagcaacgacagattcttacgtgcattagctcgcttccggggatctaatagcacgaagcttactaaaaaggagcgga

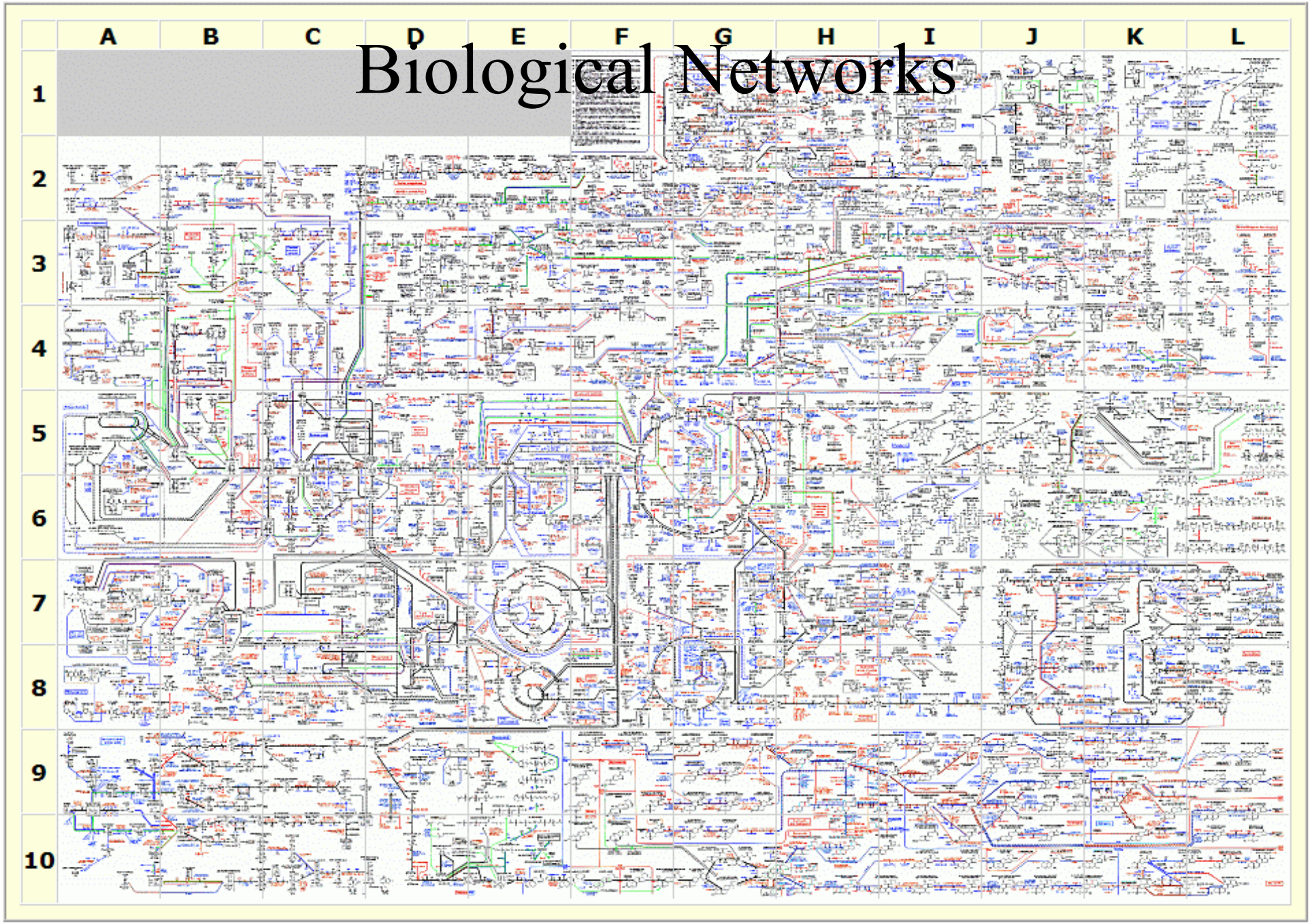
Why Finding (15,4) Motif is Difficult?

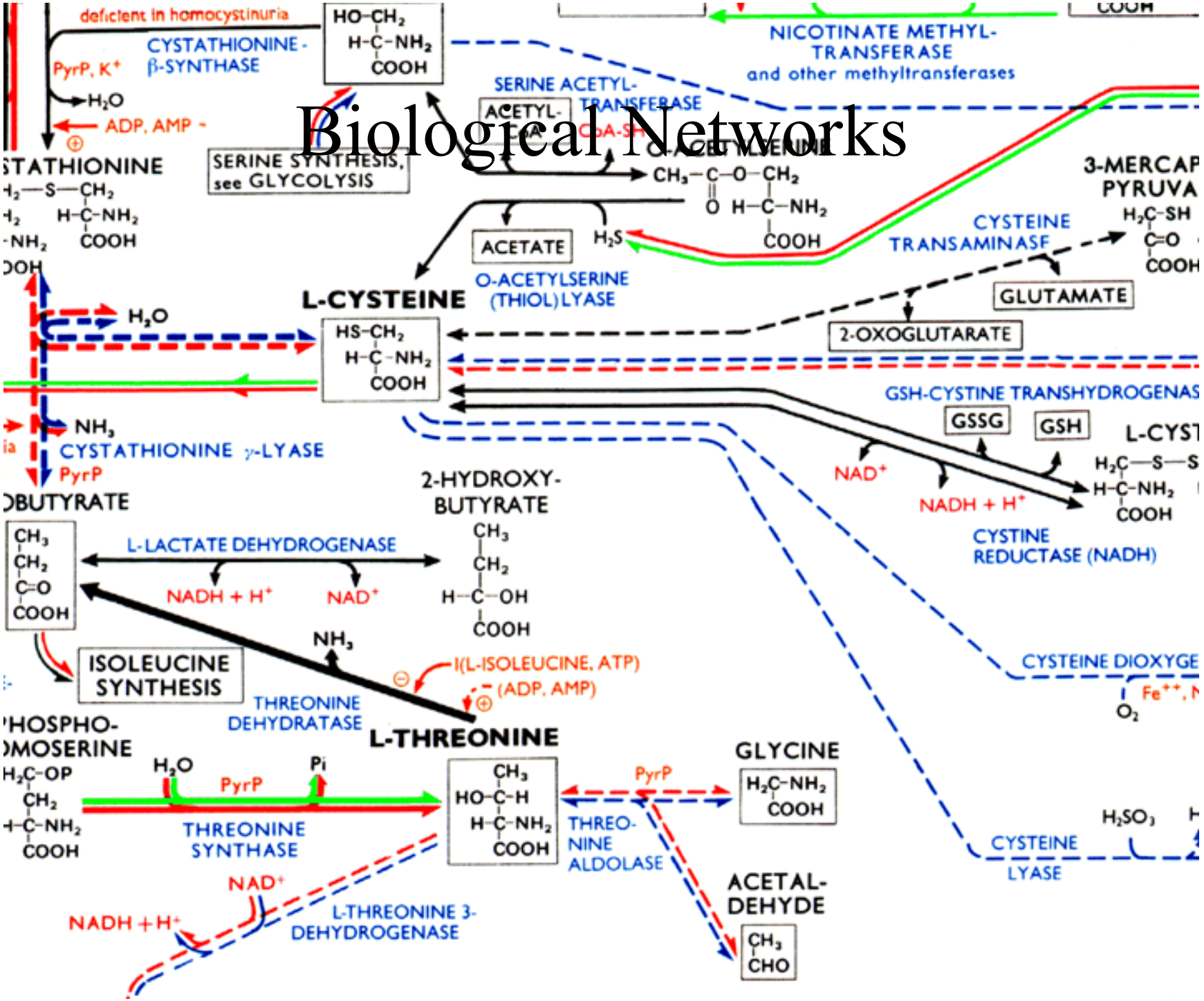


Biological Networks



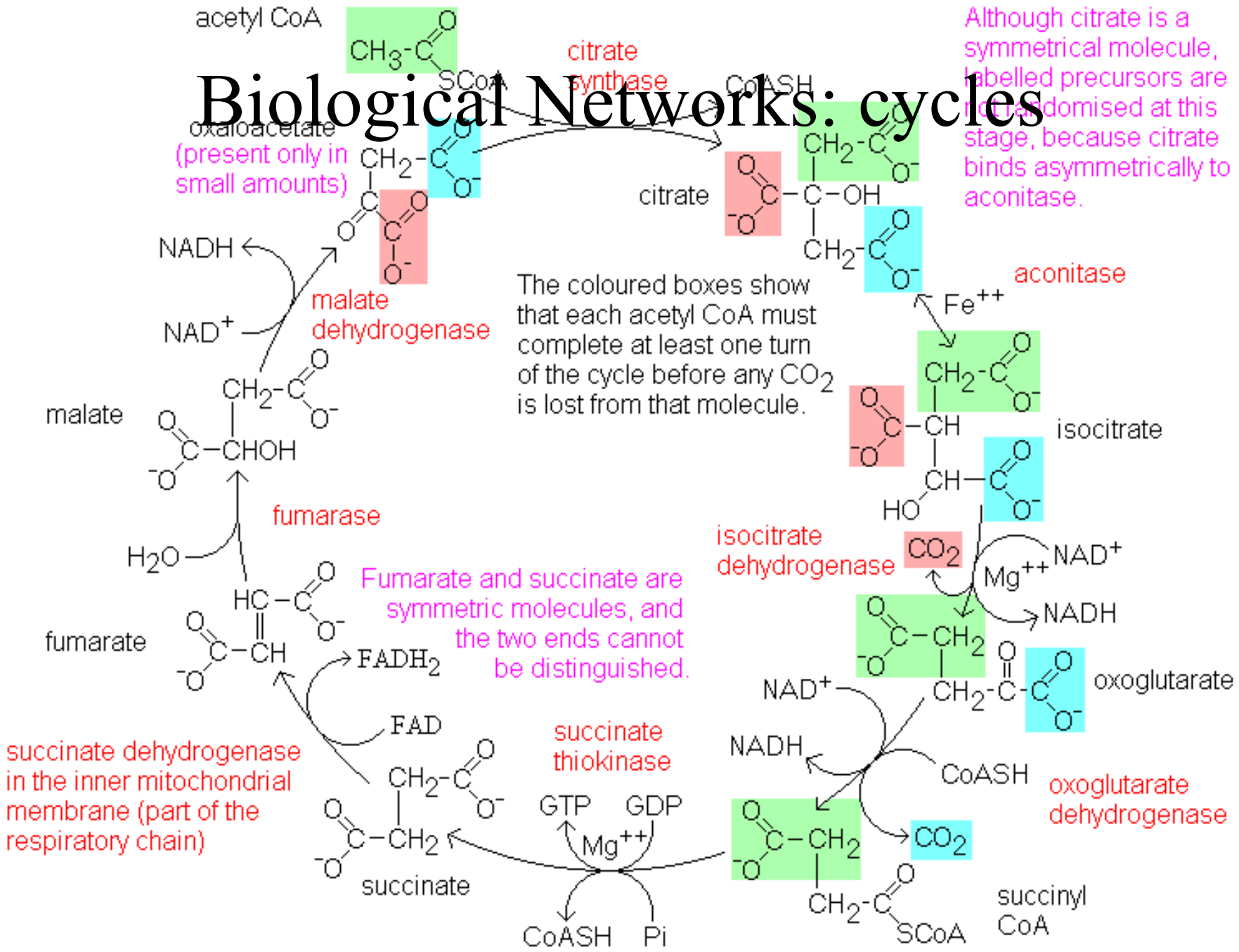
Biological Networks





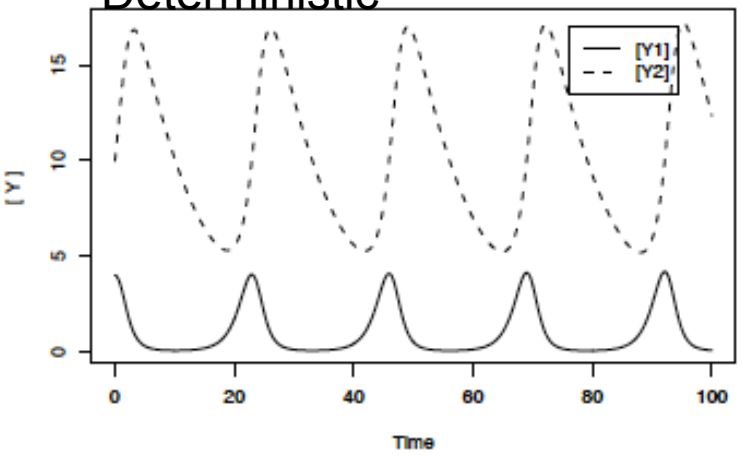
Biological Networks

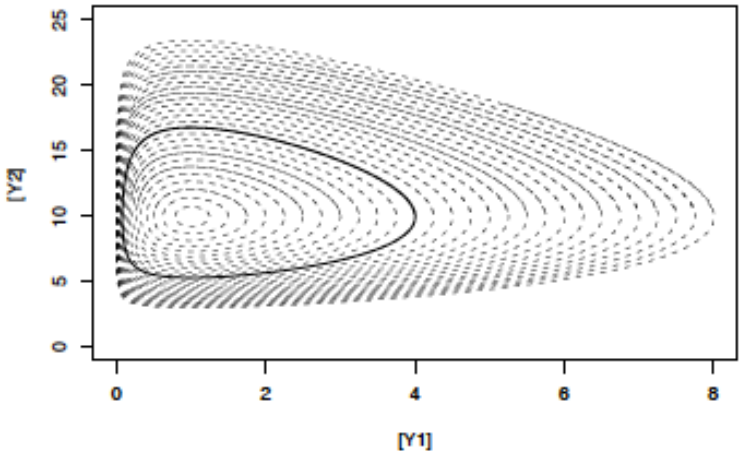
Biological Networks: cycles



Stochastic versus Deterministic

Deterministic





Stochastic