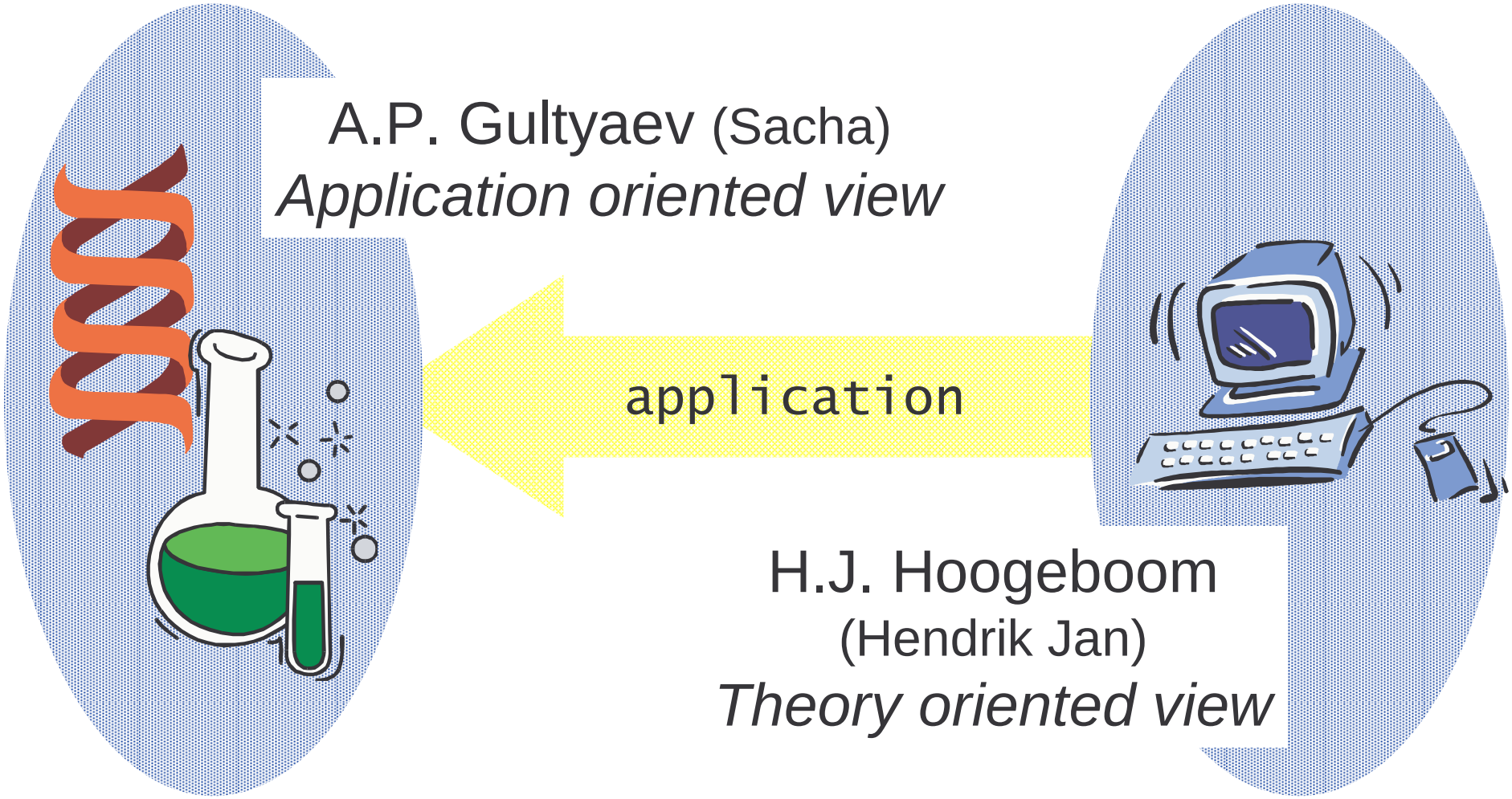
Computational Biology



A.P. Gultyaev (Sacha)

Application oriented view

H.J. Hoogeboom

(Hendrik Jan)

Theory oriented view

www.liacs.nl/home/hoogeboo/mcb/

themes

problem $\Rightarrow$ model (eg. graph)

- known algorithms
- characterization

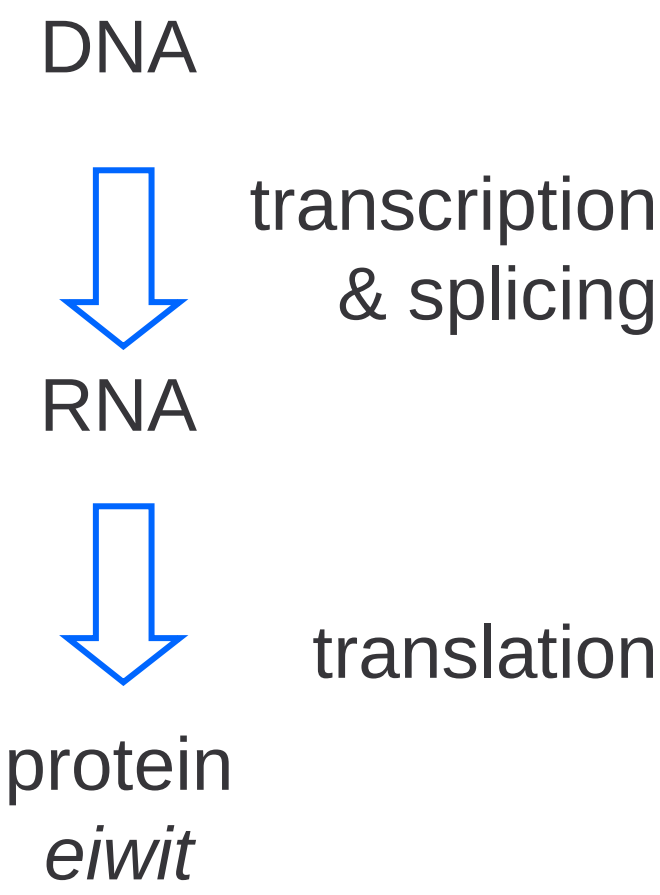
unprecise data

complexity

$\Rightarrow$ heuristics

what *is* the right answer?

central dogma



‘gene expression’

two alphabets

DNA

bases

4 symbols

a c t g

RNA

a c u g

proteins

amino acids

20 symbols

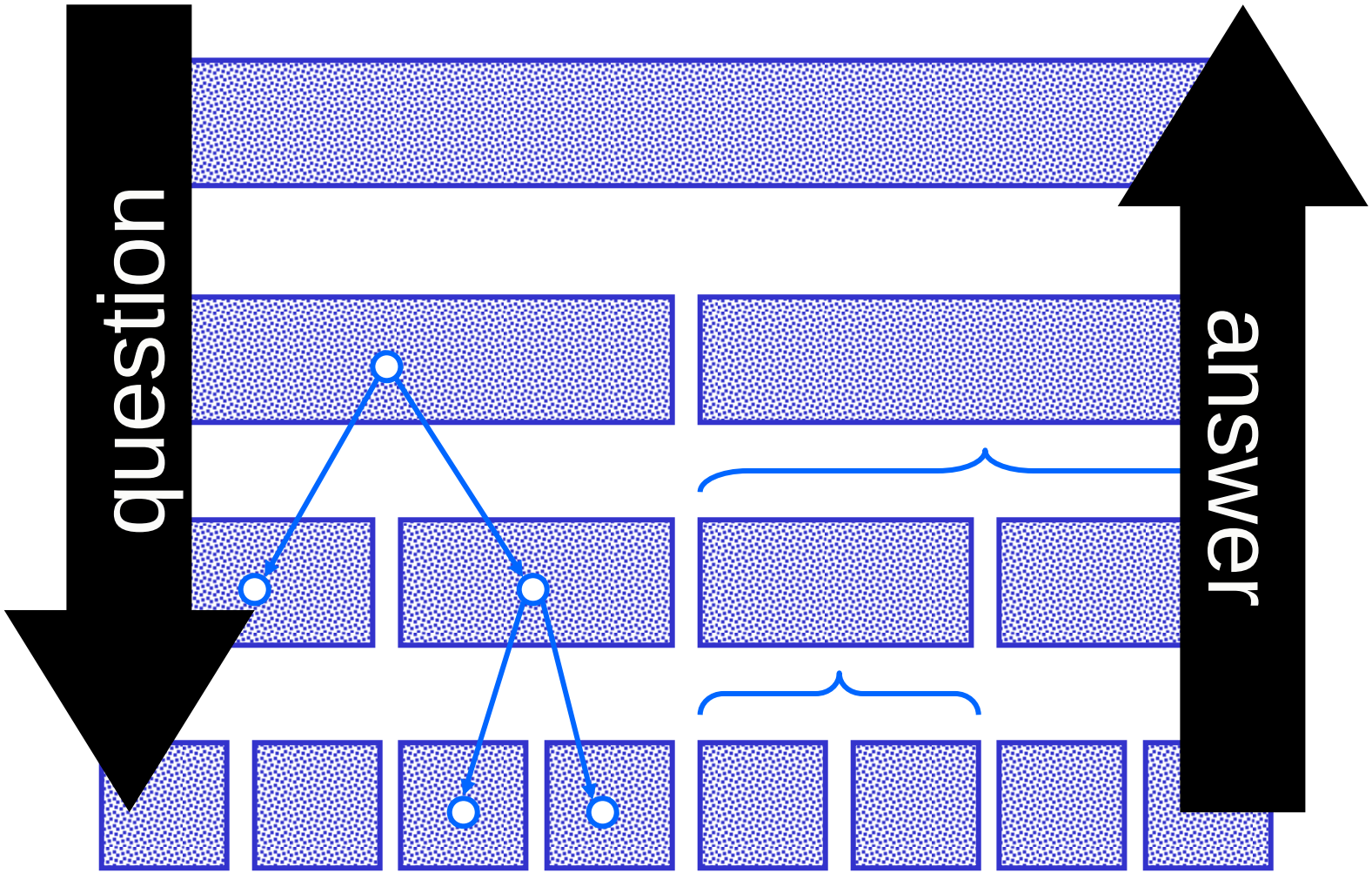
A R D N C

E Q G H I

L K M F P

S T W Y V

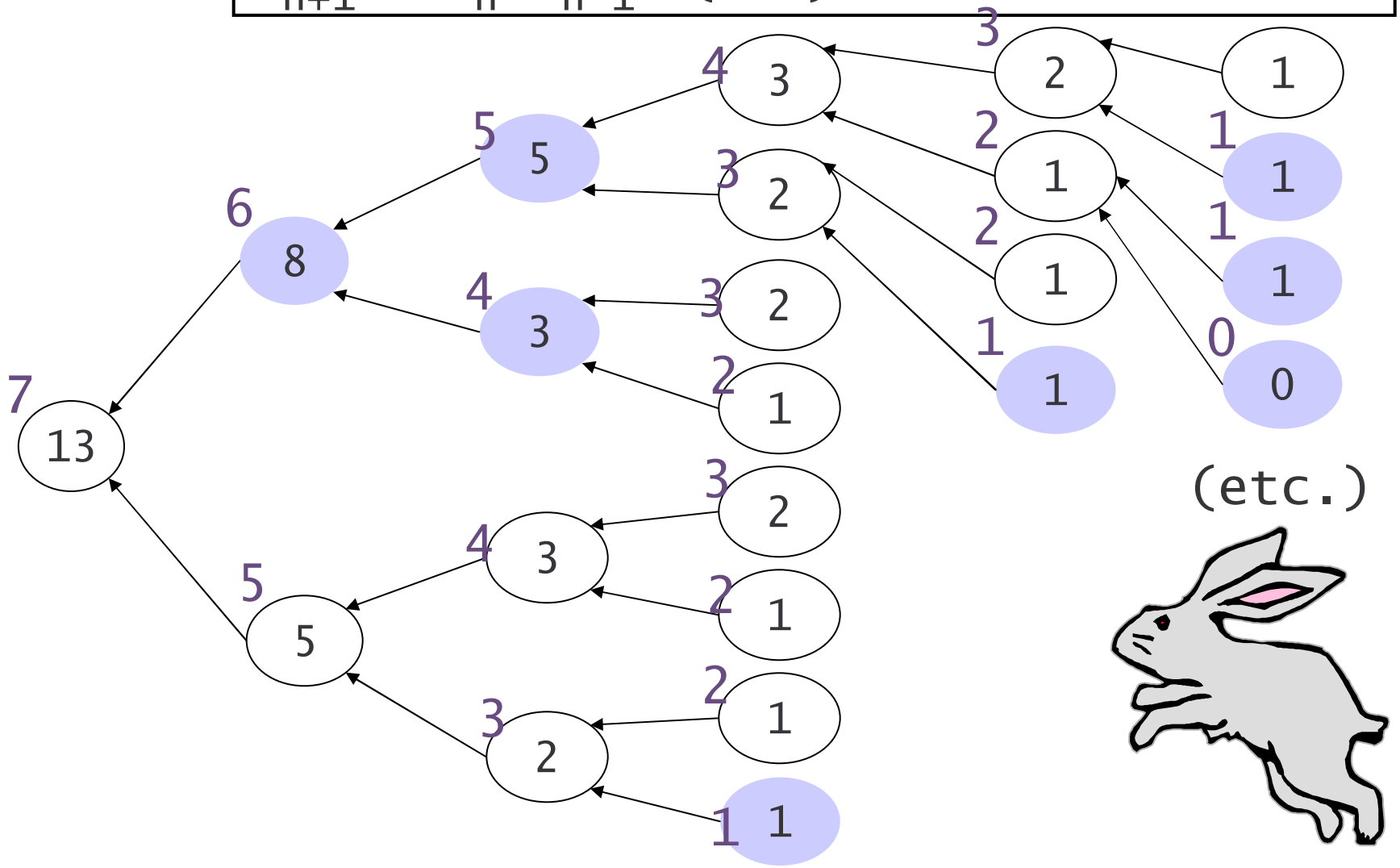
recursion



recursion: bad example

$a_0 = 0 \qquad a_1 = 1$

$a_{n+1} = a_n + a_{n-1} \quad (n \geq 1)$



recursion: dynamic programming

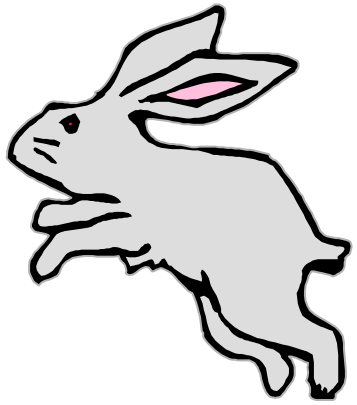
$a_0 = 0 \qquad a_1 = 1$

$a_{n+1} = a_n + a_{n-1} \quad (n \geq 1)$

0, 1, 1, 2, 3, 5, 8, 13, 21, …

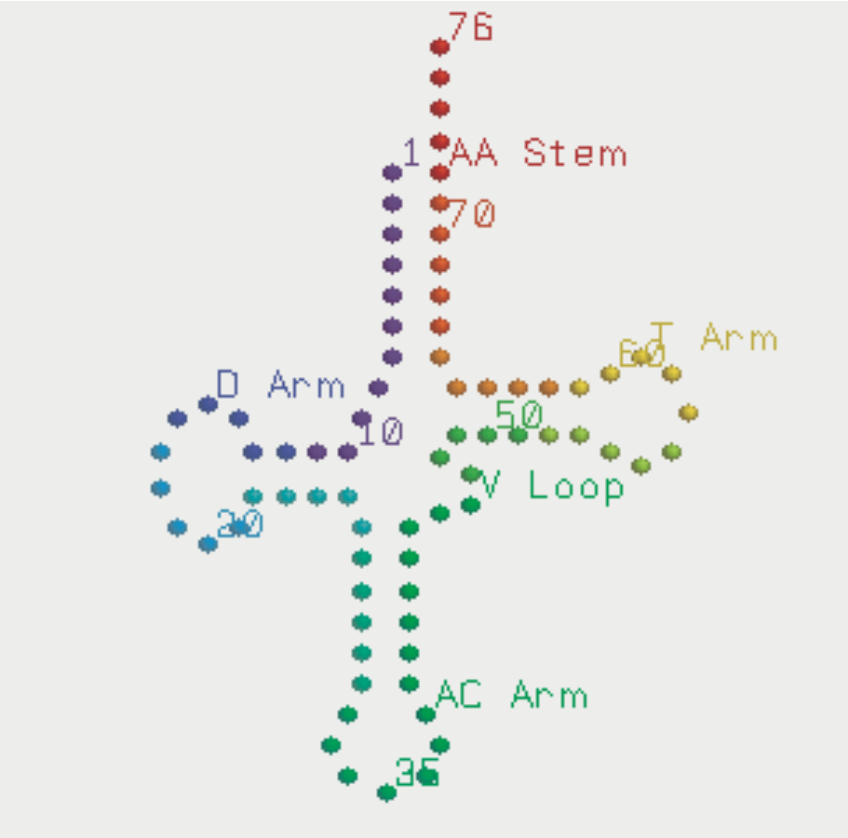
dynamic programming

- *memoization*
  store results in table
- (work bottom-up)



structure of tRNA

sequence ⇒ structure ⇒ function



2D structure

3D structure

http://alg.csie.ncnu.edu.tw/course/biology/slide_biology.ppt

sequence alignment

TCAGACGATTG
TCGGAGCTG

TCAG-ACG-ATTG
TC-GGA-GC-T-G

insertion
deletion
substitution

TCAGACGATTG
TCGGAGC--TG

match
mismatch
gap

TCAG-ACGATTG
TC-GGA-GCT-G

sequence alignment

TCAGACGATTG
TCGGAGCTG

TCAG-ACG-ATTG
TC-GGA-GC-T-G

14-6=8

TCAGACGATTG
TCGGAGC--TG

12-3-2=7

match +2
mismatch -1
gap -1

TCAG-ACGATTG
TC-GGA-GCT-G

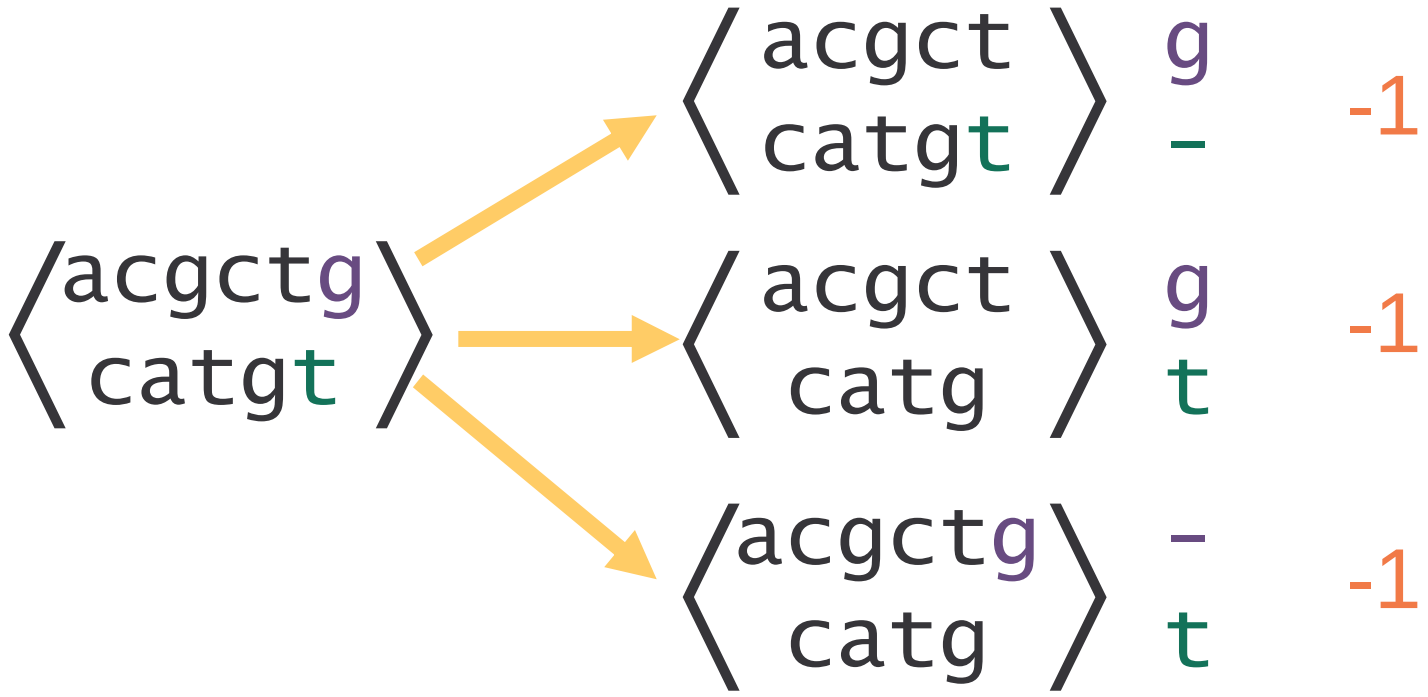
14-1-4=9

basic algorithm

alignment

- recursive principle
- dynamic programming

alignment: recursion



σ(-,x) = -1
σ(x,-) = -1
σ(x,y) = -1
σ(x,x) = 2

'reward and punishment'

dynamic programming

⟨ acgct
  cat ⟩

	-	c	a	t	g	t
-						
a						
c						
g						
c						
t				*		
g						

⟨ acgctg
  catgt ⟩

dynamic programming

⟨ acgct
  cat ⟩

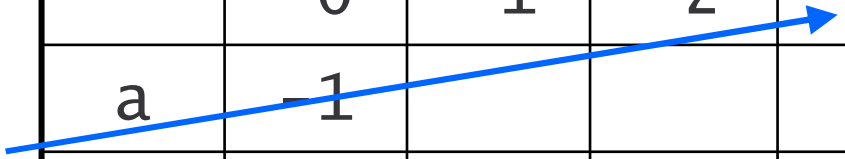
	-	c	a	t	g	t
-	0	-1	-2	-3	-4	-5
a	-1	-1	1	0	-1	-2
c	-2	1	0	0	-1	-2
g	-3	0	0	-1	2	1
c	-4	-1	-1	-1	1	1
t	-5	-2	-2	1	0	3
g	-6	-3	-3	0	3	2

⟨ acgctg
  catgt ⟩

initialization

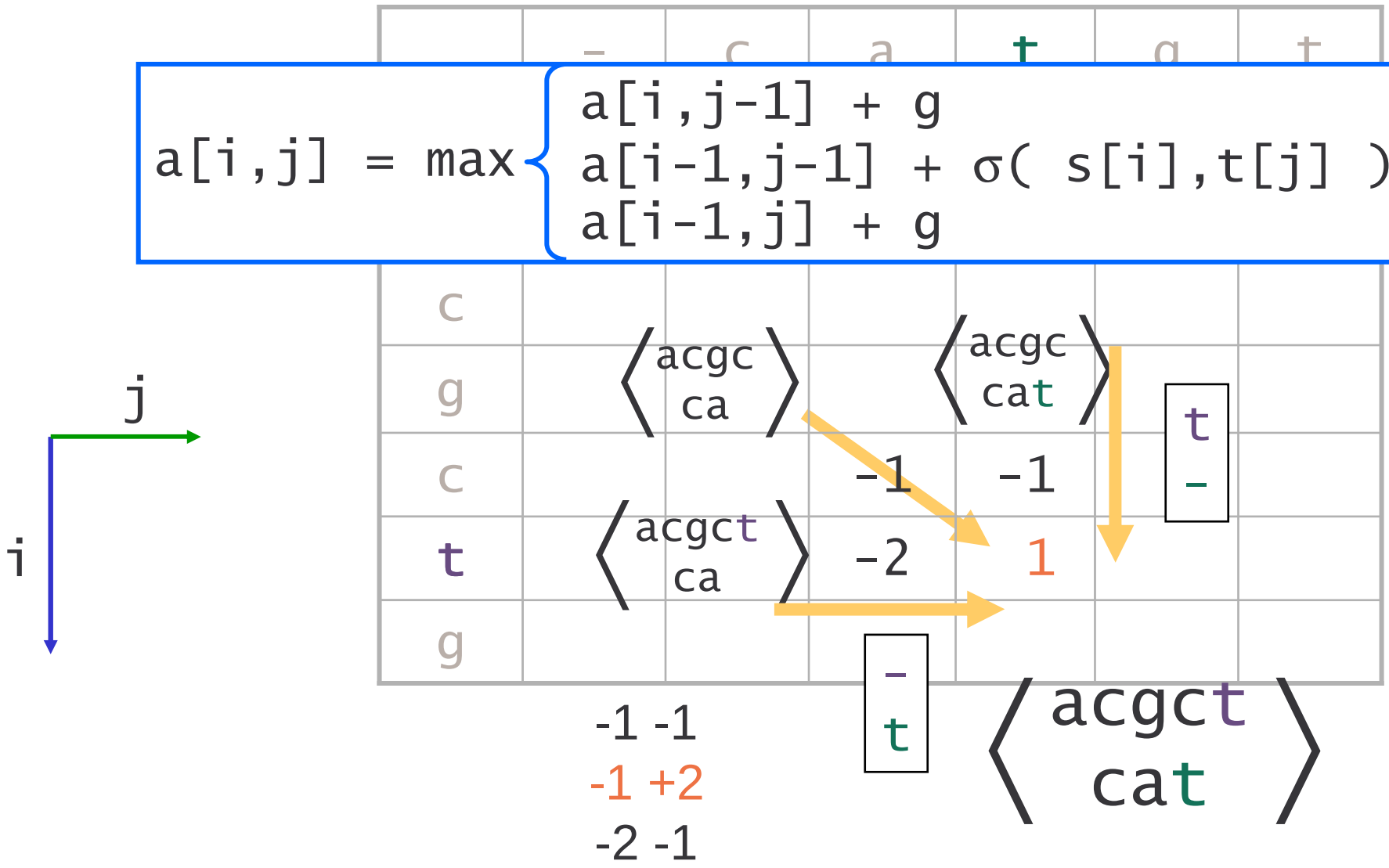
⟨ ---
cat ⟩

	-	c	a	t	g	t
-	0	-1	-2	-3	-4	-5
a	-1					
c	-2					
g	-3					
c	-4					
t	-5					
g	-6					



dynamic programming

$$a[i,j] = \max \begin{cases} a[i,j-1] + g \\ a[i-1,j-1] + \sigma(s[i],t[j]) \\ a[i-1,j] + g \end{cases}$$



alignment

	-	c	a	t	g	t
-	0	-1	-2	-3	-4	-5
a	-1	-1	1	0	-1	-2
c	-2	1	0	0	-1	-2
g	-3	0	0	-1	2	1
c	-4	-1	-1	-1	1	1
t	-5	-2	-2	1	0	3
g	-6	-3	-3	0	3	2

⟨ acgctg
  catgt ⟩

traceback

-acgctg
catg-t-

acgctg-
-ca-tgt

acgctg-
-c-atgt

⟨acgctg
 catgt⟩

	-	c	a	t	g	t
-	0	-1	-2	-3	-4	-5
a	-1	-1	1	0	-1	-2
c	-2	1	0	0	-1	-2
g	-3	0	0	-1	2	1
c	-4	-1	-1	-1	1	1
t	-5	-2	-2	1	0	3
g	-6	-3	-3	0	3	2

g
-

-
t

Eddy S.R. (2004a)

```
/* Recursion: the heart of the DP algorithm.*/
/* Initialization. */

S[0][0] = 0;
for (i = 1; i <= M; i++)  S[i][0] = i * INDEL;
for (j = 1; j <= N; j++)  S[0][j] = j * INDEL;

/* Dynamic programming, global alignment (Needleman/Wunsch) recursion. */

for (i = 1; i <= M; i++)
   for (j = 1; j <= N; j++)
     {                                          /* case #1: i,j are aligned */
       if (x[i] == y[j]) S[i][j] = S[i-1][j-1] + MATCH;
       else              S[i][j] = S[i-1][j-1] + MISMATCH;

      sc = S[i-1][j] + INDEL;                   /* case #2: i aligned to -  */
      if (sc > S[i][j]) S[i][j] = sc;

       sc = S[i][j-1] + INDEL;                  /* case #3: j aligned to -  */
       if (sc > S[i][j]) S[i][j] = sc;
     }

/* The result (optimal alignment score) is now in S[M][N]. */
```

scoring and parameter choice

```
A     A
A vs. C
```

```
A     A-
C vs. -C
```

```
AT     -AT
TA vs. TA-
```

match	M	+2	+2
mismatch	m	-1	-2
gap	g	-1	-1

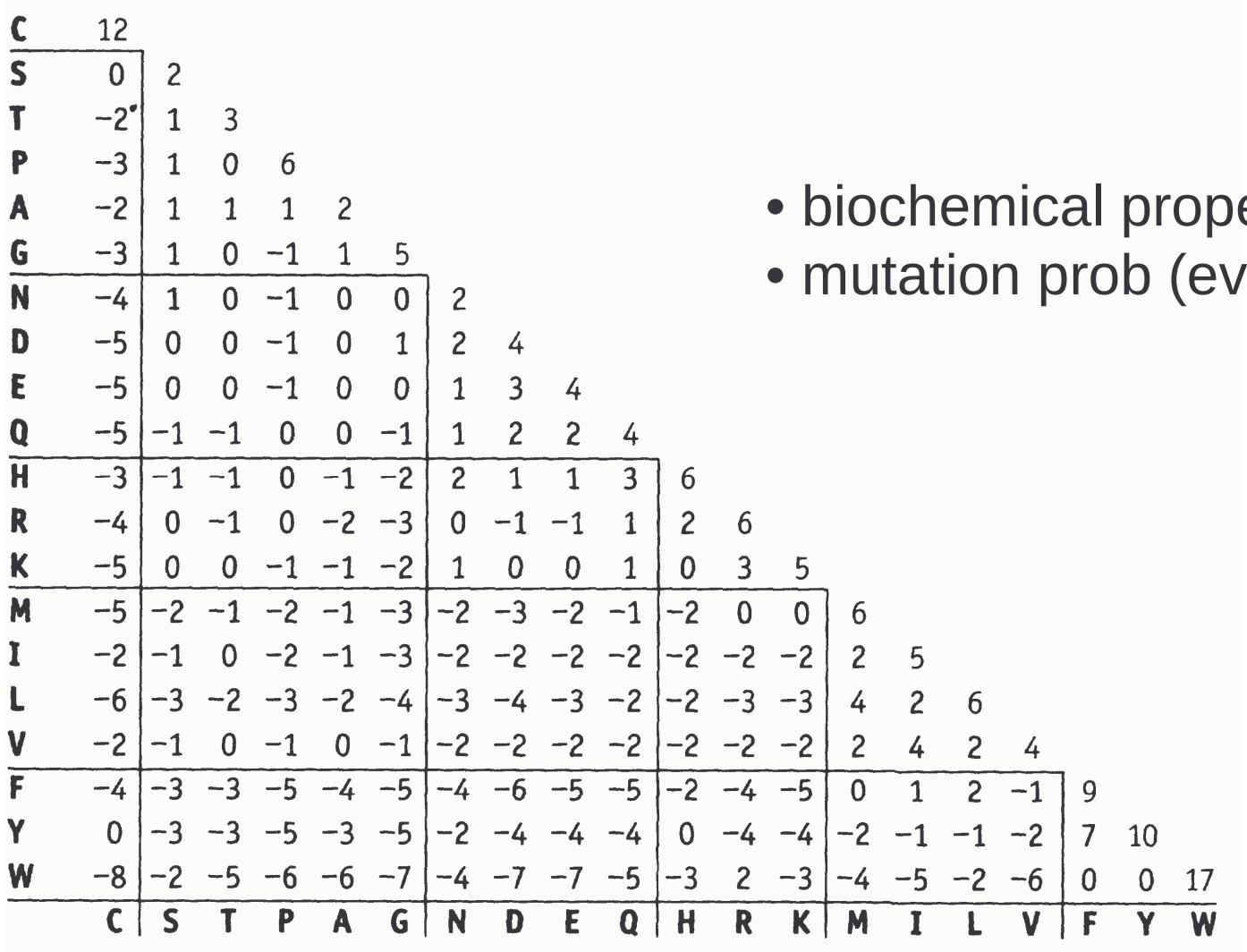
	A	C	G	T
A	91	-114	-31	-123
C		100	-125	-31
G			100	-114
T				91

gap 400 + 30k

ask your favourite molecular biologist !

PAM250 Matrix

details → notes APG



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	12																			
S	0	2																		
T	−2	1	3																	
P	−3	1	0	6																
A	−2	1	1	1	2															
G	−3	1	0	−1	1	5														
N	−4	1	0	−1	0	0	2													
D	−5	0	0	−1	0	1	2	4												
E	−5	0	0	−1	0	0	1	3	4											
Q	−5	−1	−1	0	0	−1	1	2	2	4										
H	−3	−1	−1	0	−1	−2	2	1	1	3	6									
R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	6								
K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5							
M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6						
I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5					
L	−6	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	6				
V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4			
F	−4	−3	−3	−5	−4	−5	−4	−6	−5	−5	−2	−4	−5	0	1	2	−1	9		
Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10	
W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17

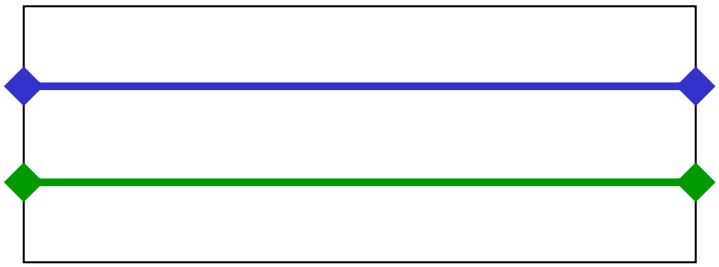
- biochemical properties
- mutation prob (evolution)

alignment

CAGCACTTGGATTCTCGG
CAGC-----G-T----GG

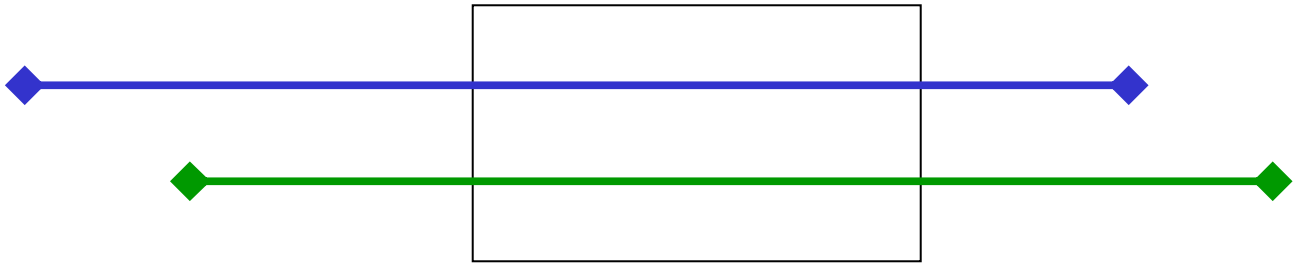
CAGCA-CTTGGATTCTCGG
---CAGCGTGG--------

variants of alignment



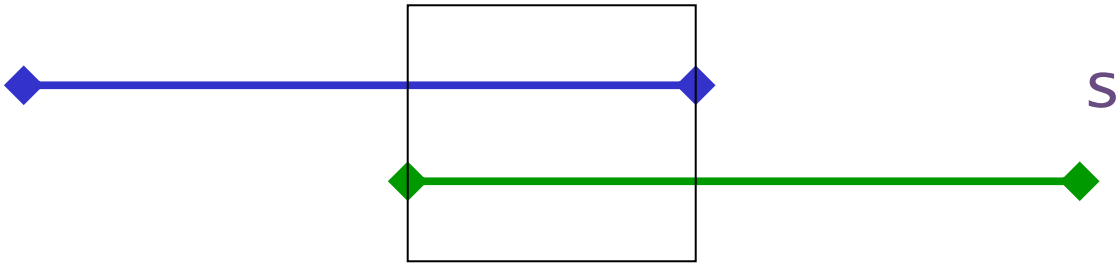
> global

Needleman & Wunsch



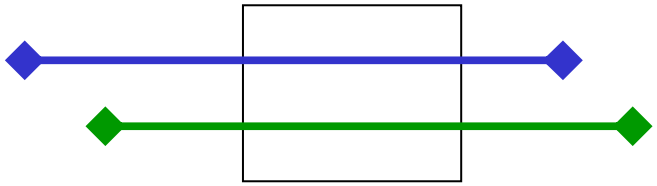
local

Smith & Waterman



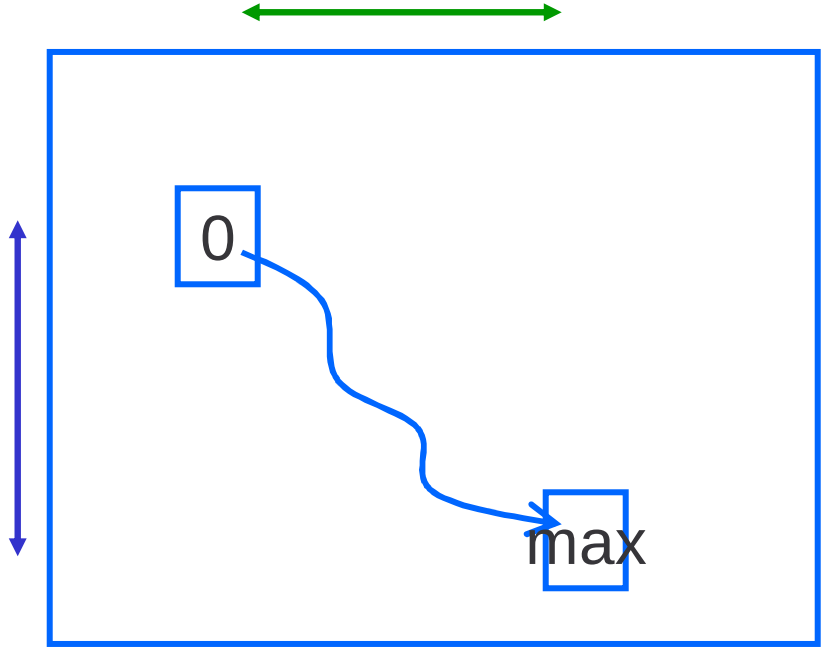
semi-global

end

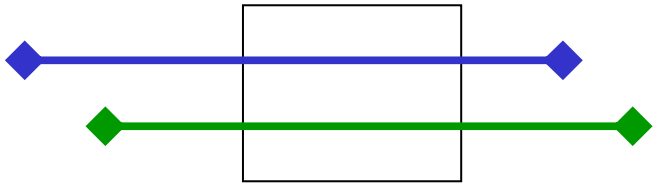


local alignment

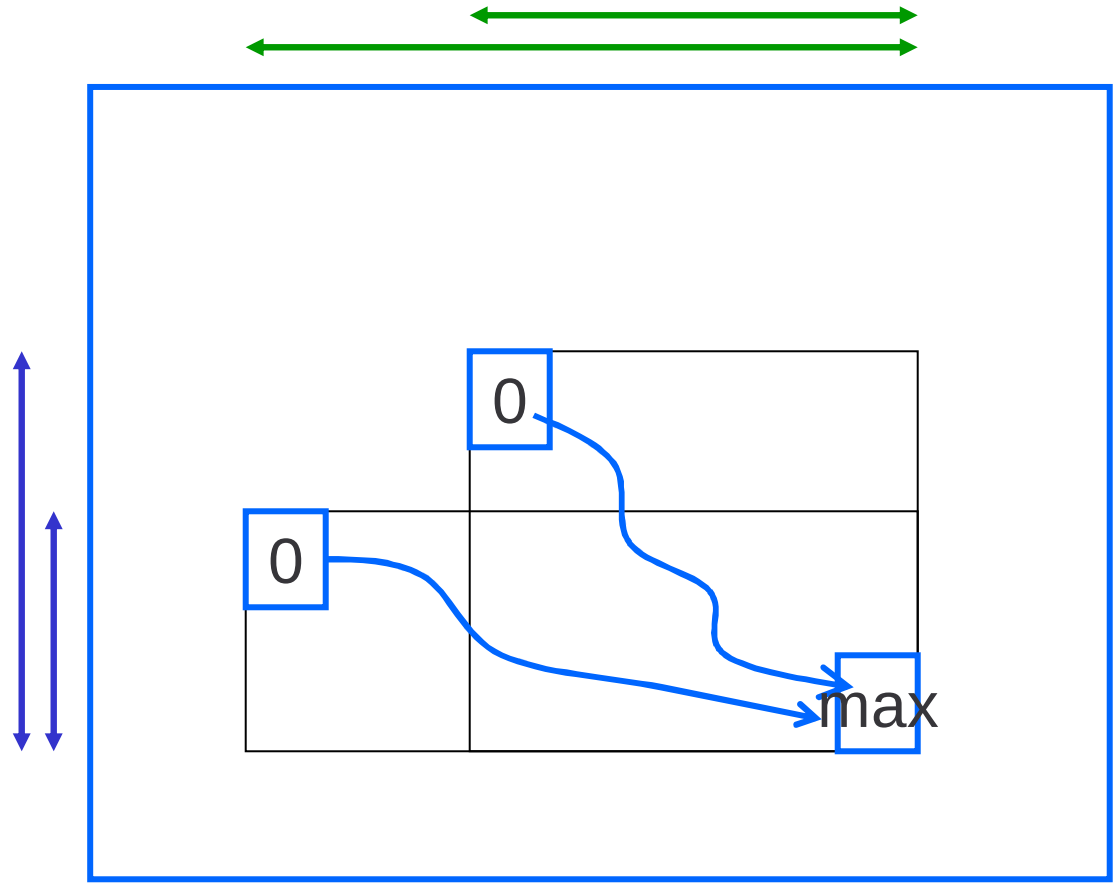
$$a[i,j] = \max \begin{cases} a[i,j-1] + g \\ a[i,j] + \sigma(\, s[i], t[j] \,) \\ a[i-1,j] + g \\ 0 \end{cases}$$

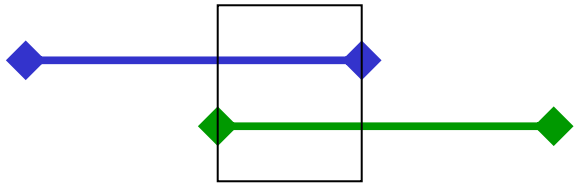


solution (*traceback*)
from max to 0

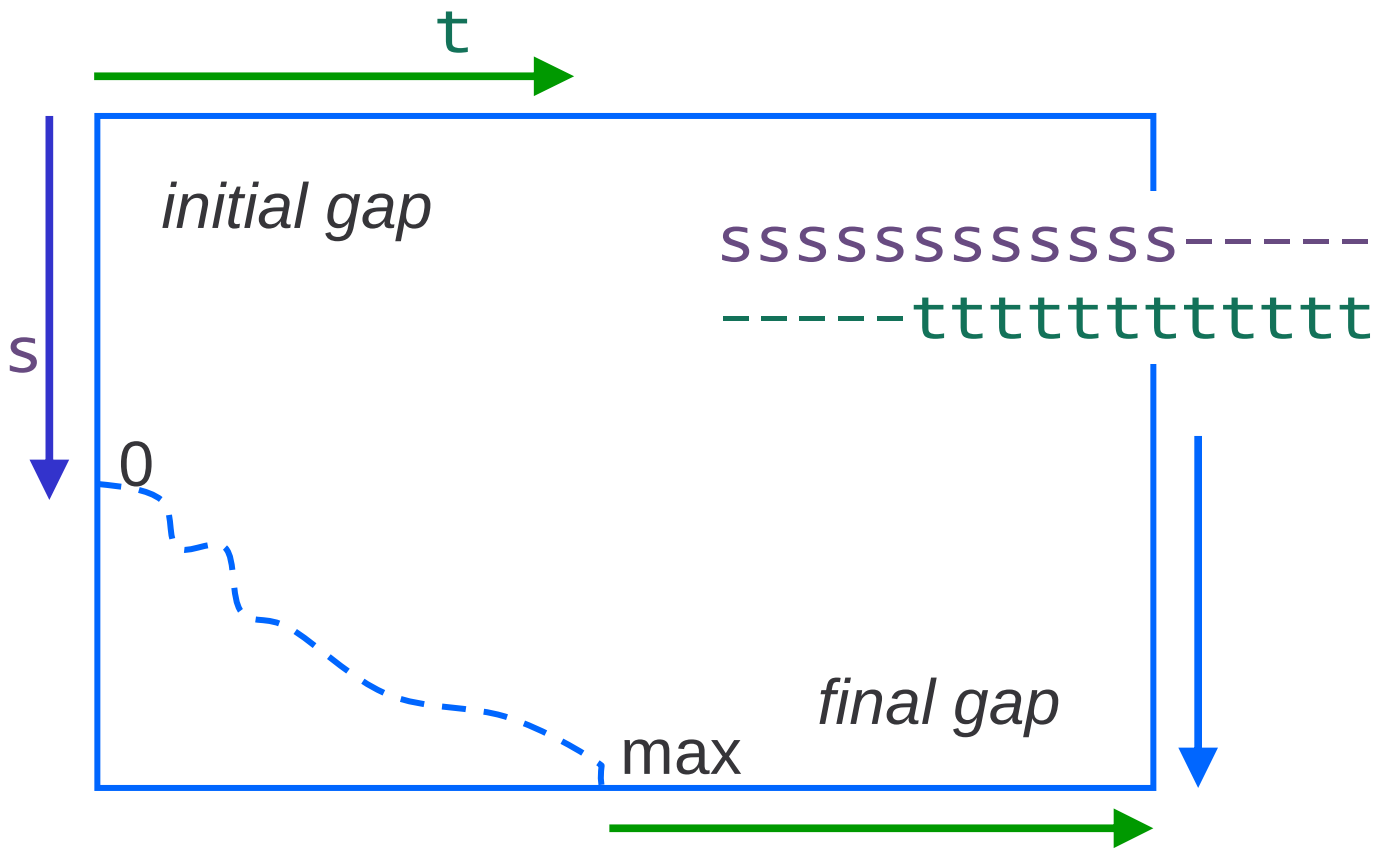


local alignment





end alignment



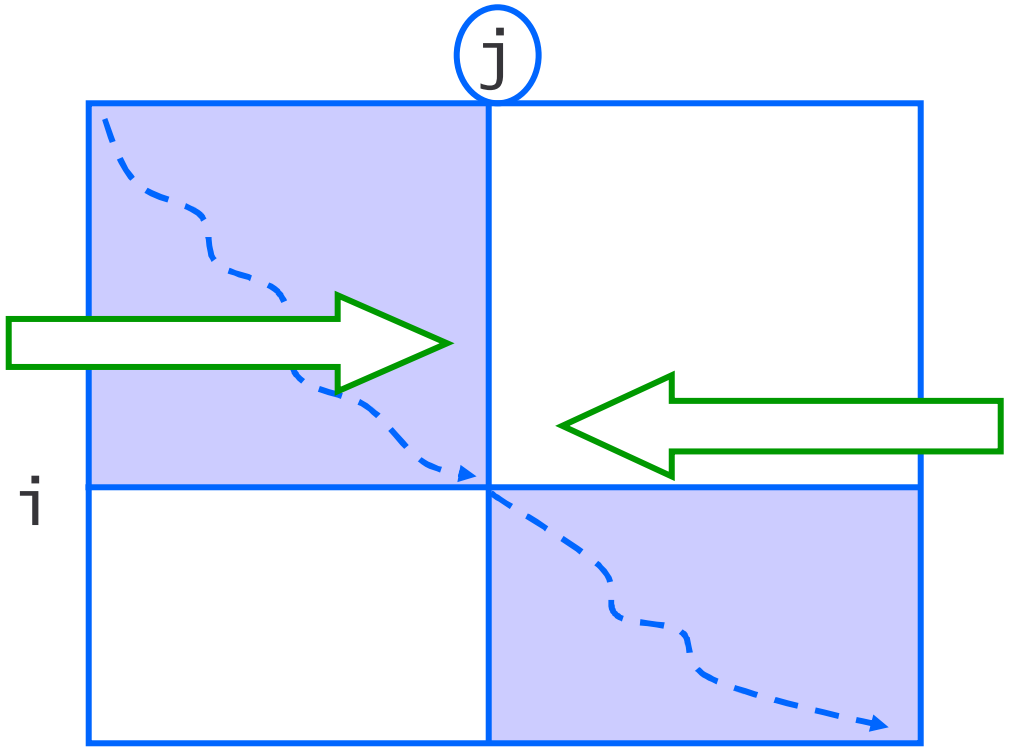
initialization with zero’s
max on border(s)

saving space

O(mn) both space & time

Hirschberg: linear space

$$\left\langle \begin{matrix} [1\ldots i\text{-}1] \\ [1\ldots j\text{-}1] \end{matrix} \right\rangle \left\langle \begin{matrix} [i\ldots m] \\ [j\ldots n] \end{matrix} \right\rangle$$



$T(m,n) \leq 2mn$

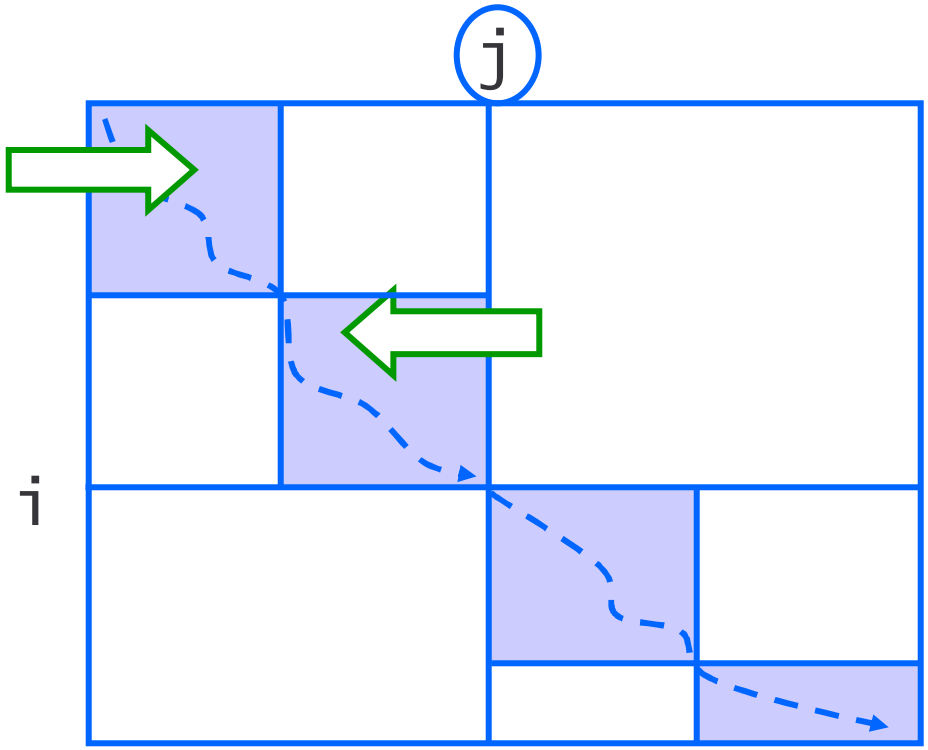
$$T(m,n) \leq mn/2 + mn/2 + T(i,n/2)+T(n-i,n/2)$$
$$\leq mn + in + (m-i)n = 2mn$$

saving space

O(mn) both space & time

Hirschberg: linear space

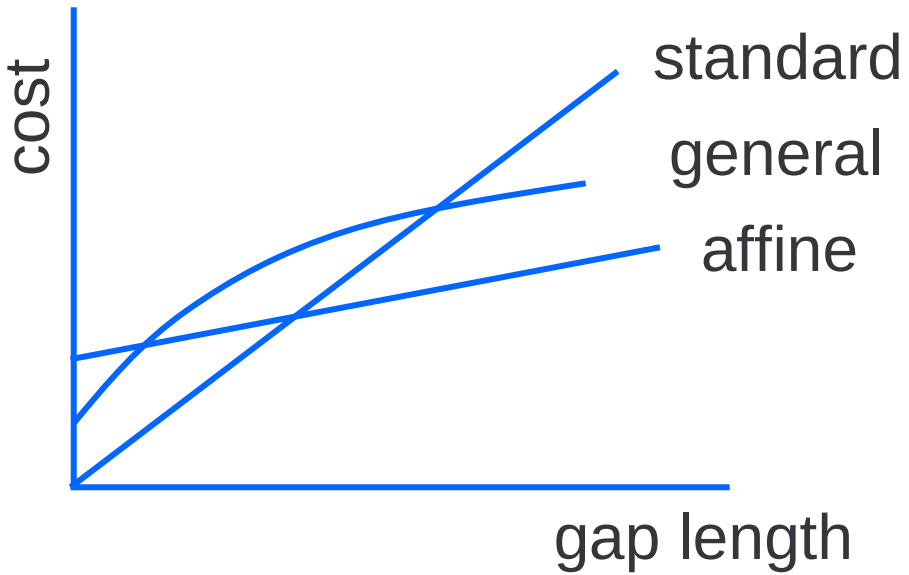
$$\left\langle \begin{matrix} [1\ldots i\text{-}1] \\ [1\ldots j\text{-}1] \end{matrix} \right\rangle \left\langle \begin{matrix} [i\ldots m] \\ [j\ldots n] \end{matrix} \right\rangle$$



1+1/2+1/4+ … = 2

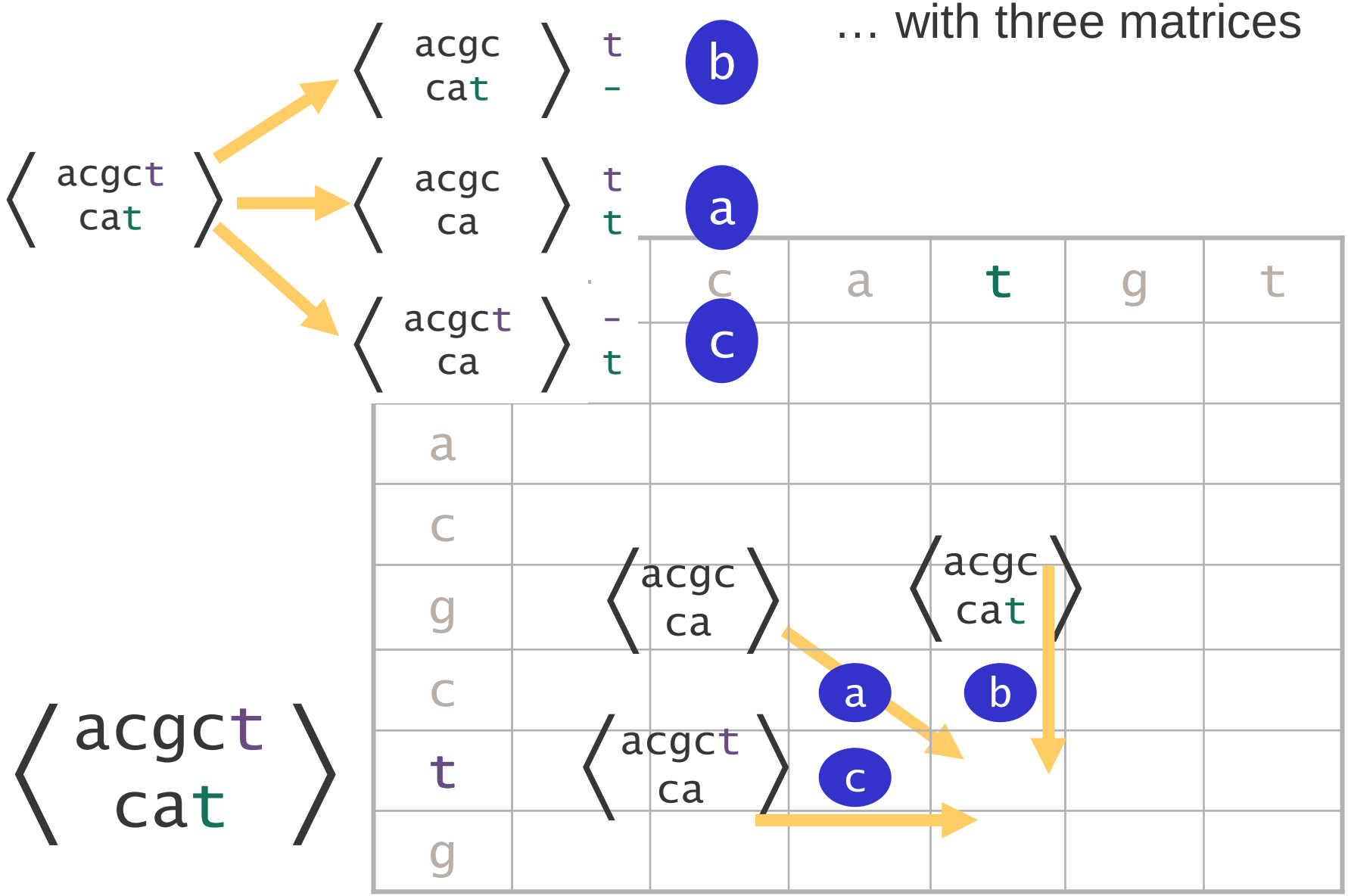
general gap penalties

g(k) =	kg	gap penalty
g(k) =	g+ke	open gap, extend gap
g(k)		arbitrary



dynamic programming

… with three matrices



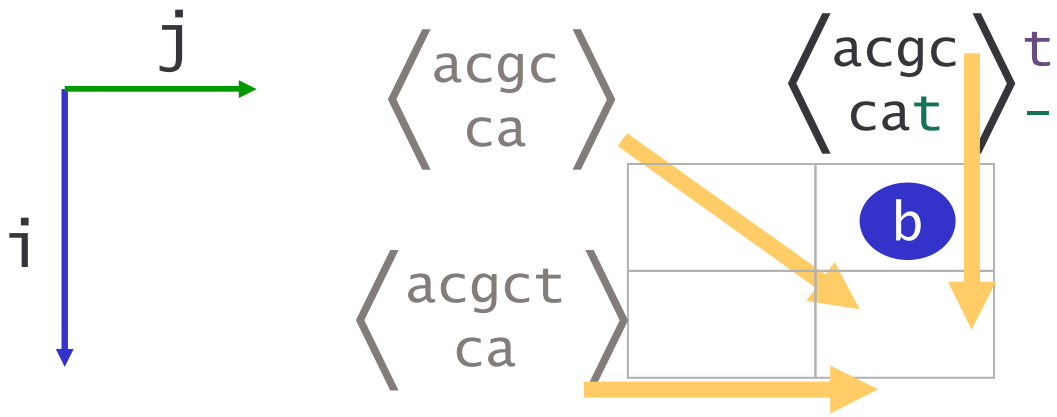
affine gap penalties

dynamic programming
with three matrices

$g(k) = g+ke$ open gap, extend gap

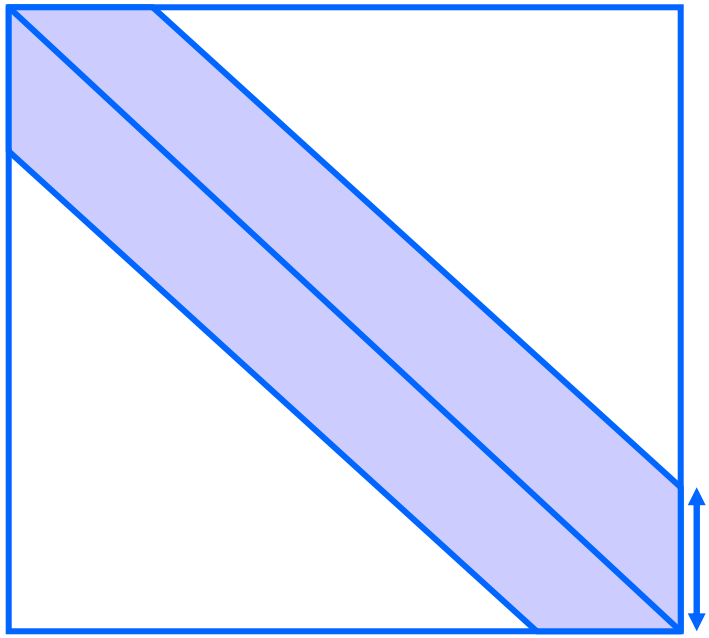
$$b[i,j] = \max \begin{cases} a[i-1,j] - g - e \\ b[i-1,j] - e \\ c[i-1,j] - g - e \end{cases}$$

← *gap was already open*



O(mn) vs.
$O(mn^2 + m^2n)$
general gap penalty

comparing similar sequences

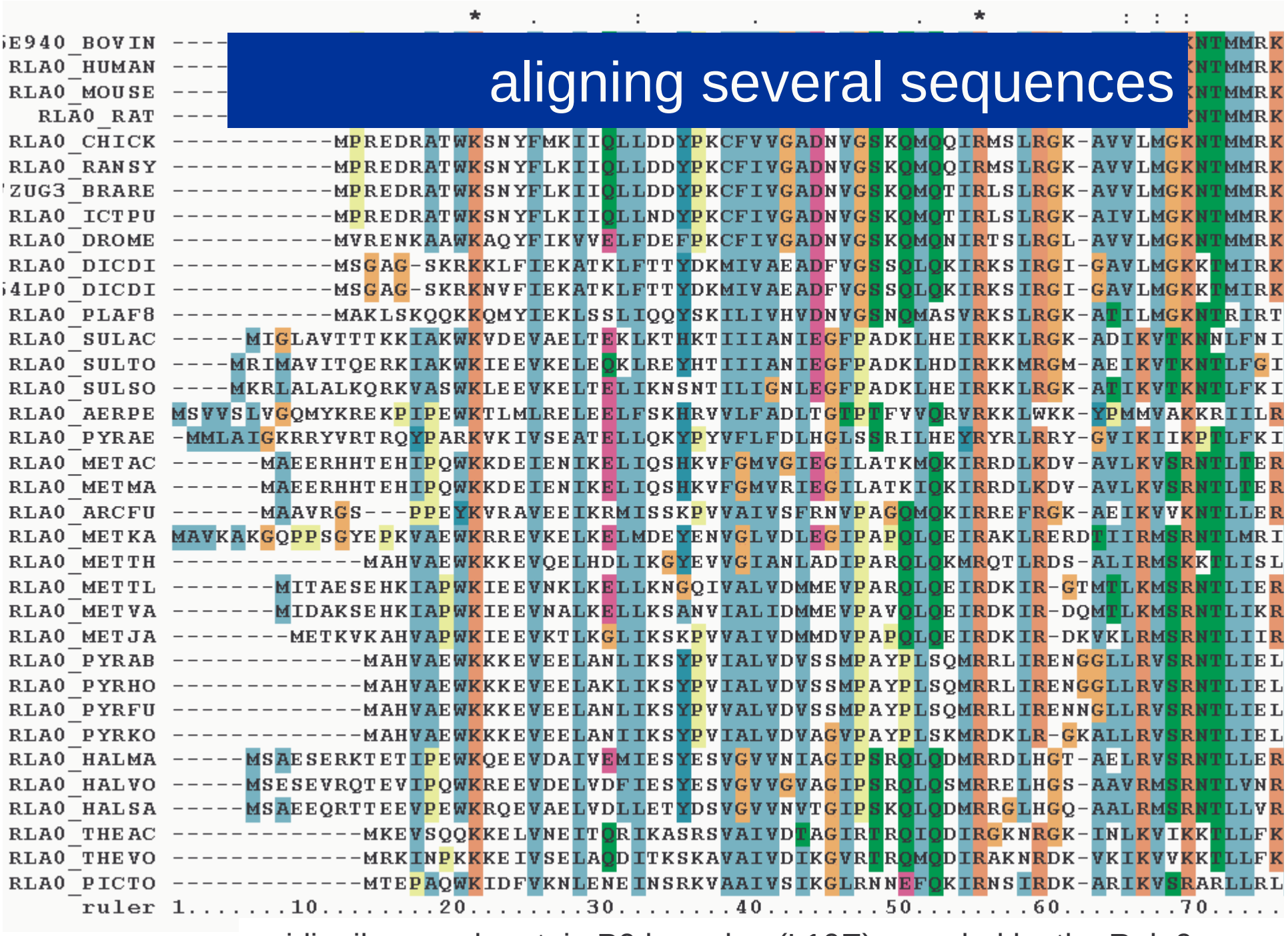


O(mn) *quadratic*
both space & time
reduce time

k depends on
'number of dashes'

banded alignment
linear time

- heuristic
- set k based on upper bound for alignment
- exact → while improvement do k = 2k



acidic ribosomal protein P0 homolog (L10E) encoded by the Rplp0 gene

comparing multiple sequences

details → notes APG

how to score?

SP sum-of-pairs

- pairs of symbols
- pairs of strings

```
A – C T G – G G
A A – T G – C G
A A C T C C C –
```

dynamic programming

possible … but too much space & time

heuristic reduce search space (banded alignment)

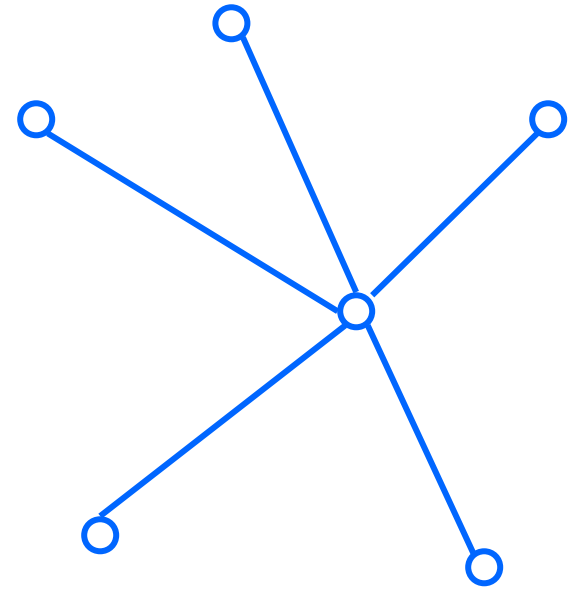
problem is NP-complete (# strings = dimension!)

heuristic

star alignment / progressive alignment

hidden Markov model

star alignment



string in center (which one?)
compute pair-wise alignments
extend pairs into multiple alignment
“once a gap, always a gap”

$O(k^2m^2)$ k sequences, m length

theory: within factor 2 of optimal alignment

star alignment

```
1 ATTGCCATT
2 ATGGCCATT
3 ATCCAATTTT
4 ATCTTCTT
5 ACTGACC
```

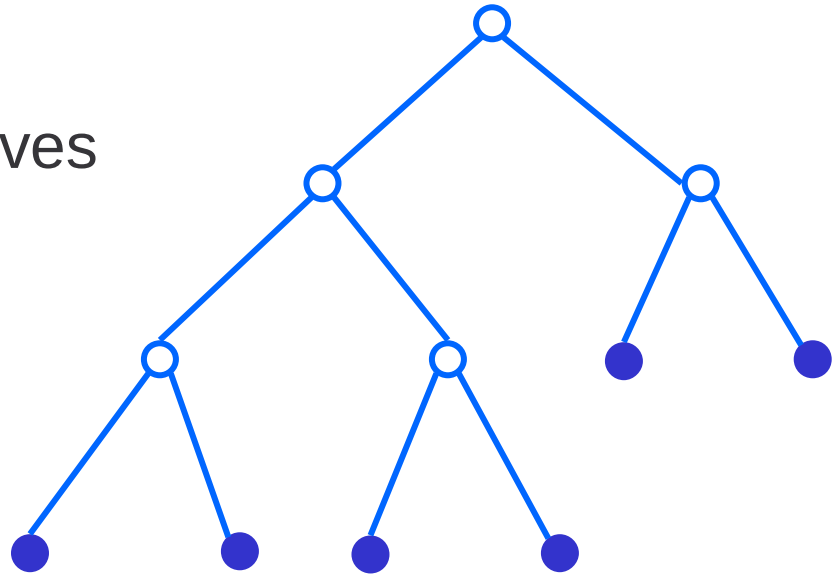
	1	2	3	4	5	Σ
1		7	-2	0	-3	2
2			-2	0	-4	1
3				0	-7	-11
4					-3	-3
5						-17

```
1 ATTGCCATT
2 ATGGCCATT

1 ATTGCCATT--
3 ATC-CAATTTT

1 ATTGCCATT
4 ATCTTC-TT

1 ATTGCCATT
5 ACTGACC--
```

```
1 ATTGCCATT--
2 ATGGCCATT--
3 ATC-CAATTTT

1 ATTGCCATT--
2 ATGGCCATT--
3 ATC-CAATTTT
4 ATCTTC-TT--
5 ACTGACC----
```

progressive alignment

guide tree
bottom-up
sequences in leaves



- tree known ← evolutionary relation
- build tree ← clustering algo's ClustalW

inner nodes
combine alignments (profiles variant basic algo)

probleem solved !?

much too slow

- long strings
- huge databases

heuristics

- **FASTA** along diagonal
- **BLAST** minimal close match

multiple alignment

(several strings)

NP complete … exponential

statistics

- 23 pairs of chromosomes
- 3.1 10^9 nucleotide bases
- avarage 3000 bases / gene
  dystrophin 2.4 million
- 30,000 - 40,000 genes
- protein variants 1 million (splicing)
- 99.9% exactly the same in humans

Basic Local Alignment Search Tools

query

query word (W=3)

GSVEDTTGSQSLAALLNKCKTPQGQRLVNQWIKQPLMDKNRIEERLNL

neighbourhood words

threshold

PQG	18
PEG	15
PRG	14
PMG	13
PQA	12
PQN	12

SLAALLNKCKTPQGQRLVNQWIKQPLMDKNRIEERLNL

TLASVLDCTVTPMGSRMLKRWLHMPVRDTRVLLERQQT

subject

high-scoring segment pair (HSP)

parameters:
W word size
T threshold
S score
E expected