Lessons 5 & 6. Content

1. Alignment algorithms

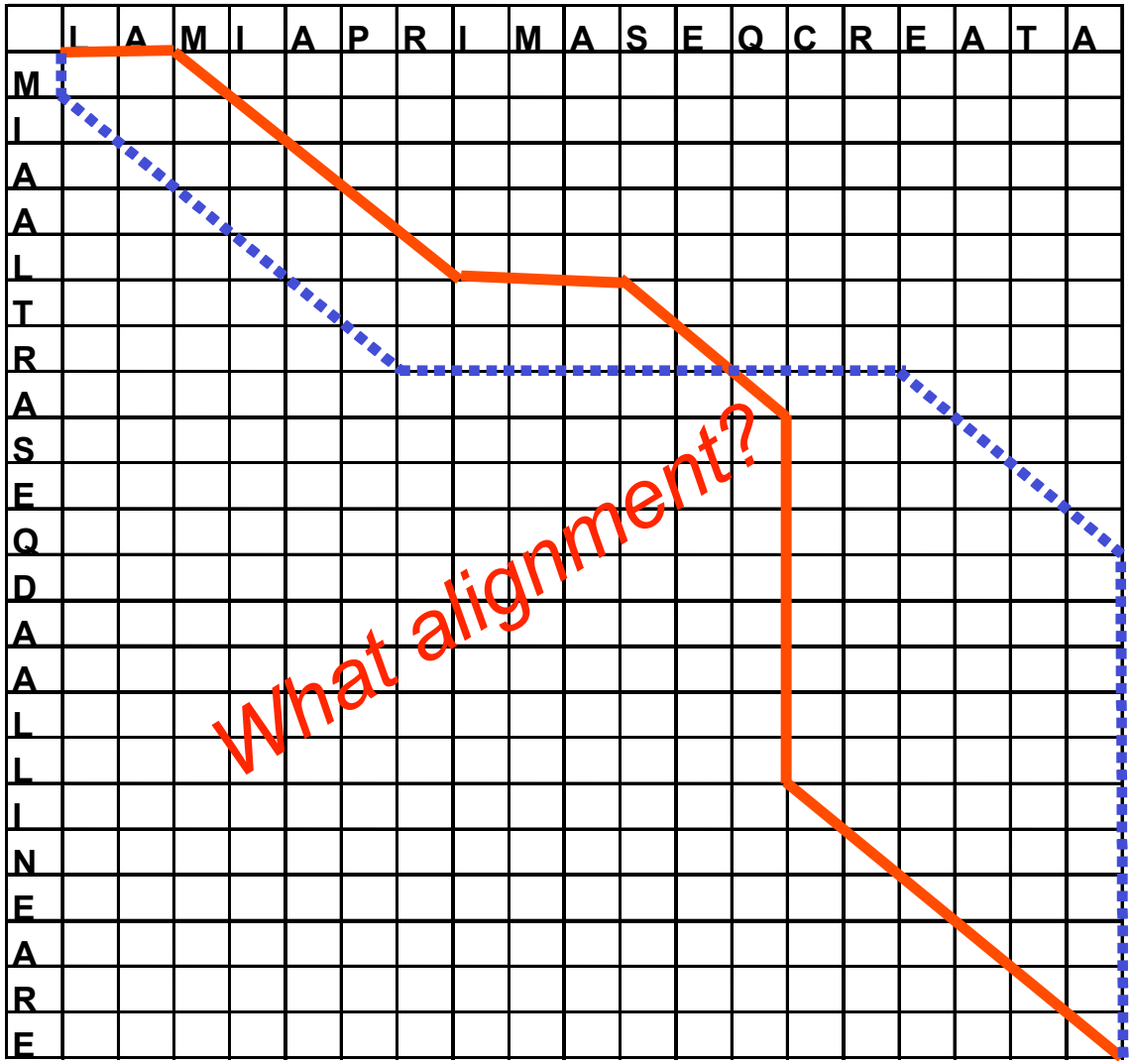
2. Multiple alignments

Alignment

1) Score for the correspondence of aa/bases
2) Penalty of insertions/deletions
3) Algorithm performing the alignment
4) Measure of the alignment significance

Alignment

1) Score for the correspondence of aa/bases
2) Penalty of insertions/deletions
3) Algorithm performing the alignment
4) Measure of the alignment significance



LAMIAPRIMASEQCREATA----------

-MIAALTR-------ASEQDAALLINEARE

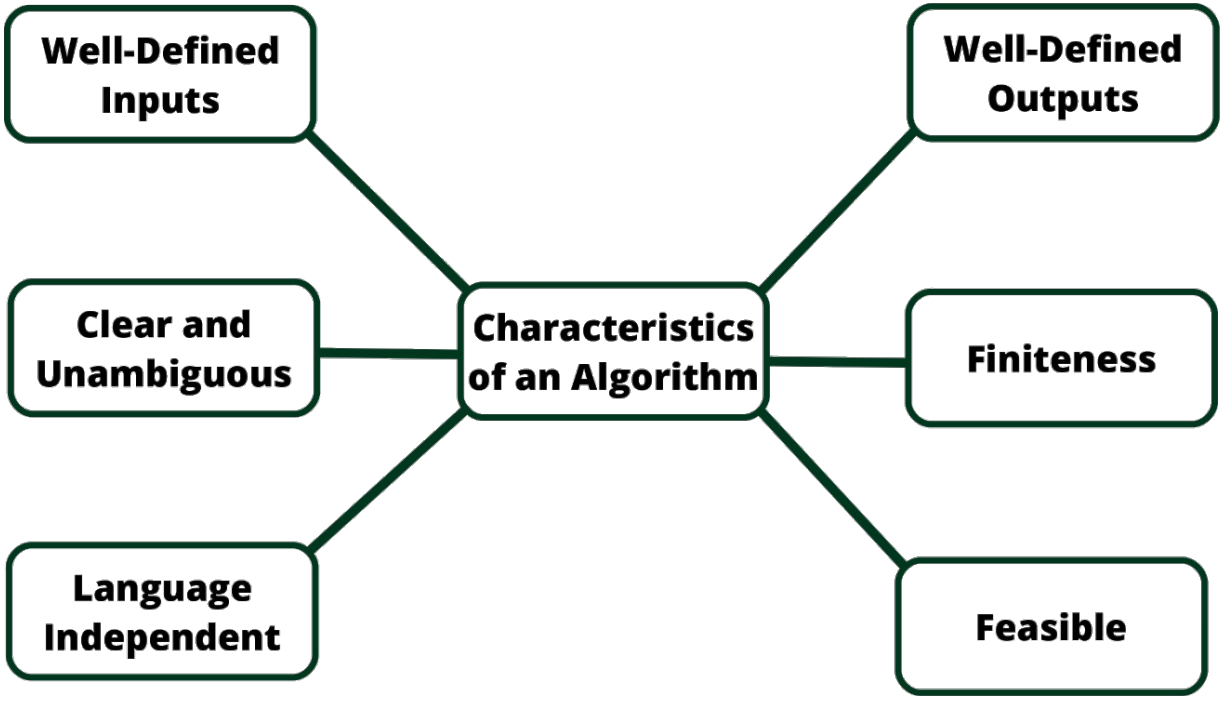
LAMIAPRIMASEQ--------CREATA

--MIAAL---TRASEQDAALLINEARE

Some definitions…

From the persian mathematician al-Kharezmi (al-Khawarizmi) from the IX century

Algorithm: Sequence of well-defined instructions (univocally interpretable) that allows to reach an outcome in a finite number of steps



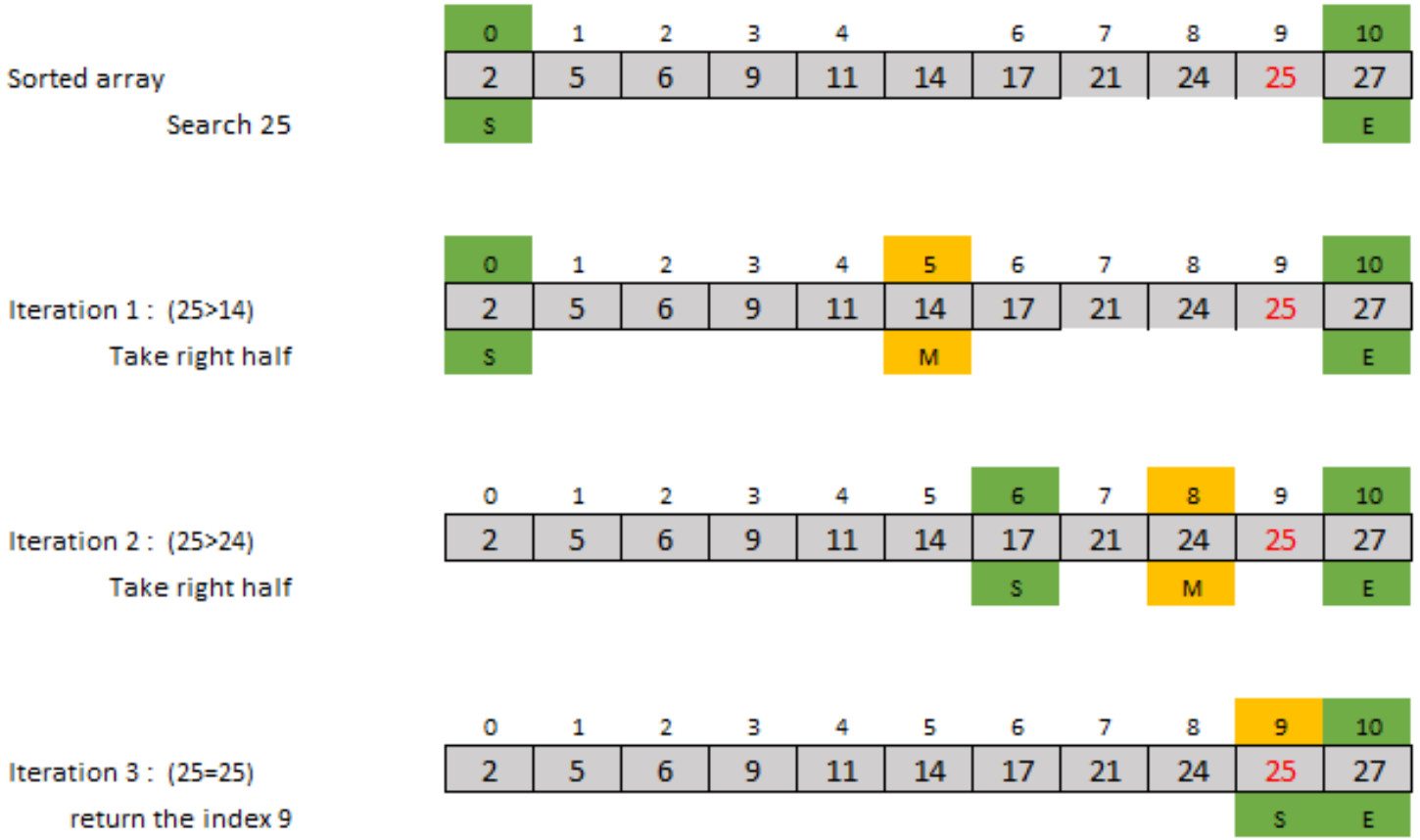
Some definitions…

From the persian mathematician al-Kharezmi (al-Khawarizmi) from the IX century

Algorithm: Sequence of well-defined instructions (univocally interpretable) that allows to reach an outcome in a finite number of steps

Step 1: Gather Your Ingredients for the Sandwich.
Step 2: Put Gloves on (optional).
Step 3: Pull Out Two Slices of Bread and put on plate.
Step 4: Open Peanut Butter and Jelly.
Step 5 Pick up butter knife.
Step 6: Spread the Peanut Butter Onto One Slice of Bread.
Step 7: Spread the Jelly Onto the Other Slice of Bread.
Step 8: Combine the Two Slices.
Step 9: Say: Enjoy

An example…



The **binary search** (or half-interval search) is an algorithm that finds the position of a target value within a sorted array.
It compares the target value to the middle element of the array; if they are not equal, the search continues on the remaining half, again taking the middle element *etc.*

Some definitions…

From the persian mathematician al-Kharezmi (al-Khawarizmi) from the IX century

Algorithm: Sequence of well-defined instructions (univocally interpretable) that allows to reach an outcome in a un risultato in a finite number of steps

Programme: description of an algorithm in a specific coding language (e.g. C, fortran, python, perl).

Exact algorithms *vs.* heuristic algorithms

Type of algorithm	programmes	Pro	Cons
Exact	1. Needlman Wunch (1970) 2. Smith Waterman (1981)	Sensitive	Slow
Heuristic	1. BLAST –Altschul (1990) 2. FASTA- Pearson (1985)	Fast	Non sensitive

Some definitions…

Heuristic method: in computer science, it is heuristic a method that can generally find a good solution to a problem, though it is not possible to prove that the found solution is the correct one

Exact algorithms:
e.g. algorithm of Smith-Waterman

Exact, it guarantees to find out the best alignament(s) for a pair of sequences.

For 2 sequences: A of length n and B of length m,
Smith-Waterman takes n*m computational steps

In searching a database of sequences -

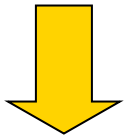
- In case the query sequence A is long n=200 nucleotides
- And we search for homologues sequences in the EST database containing, e.g., $23*10^6$ sequences, B_i, each of length m=500.
- Number of computational steps:
  $23*10^6 * 500 * 200 \sim 10^{11}$ total steps !

Exact algorithms:
e.g. algorithm of Smith-Waterman

Exact, it garantees to find out the best alignament(s) for a pair of sequences.

For 2 sequences: A of length n and B of length m,
Smith-Waterman takes n*m computational steps
$*10^6 * 500 * 200 \sim 10^{11}$ passi totali !

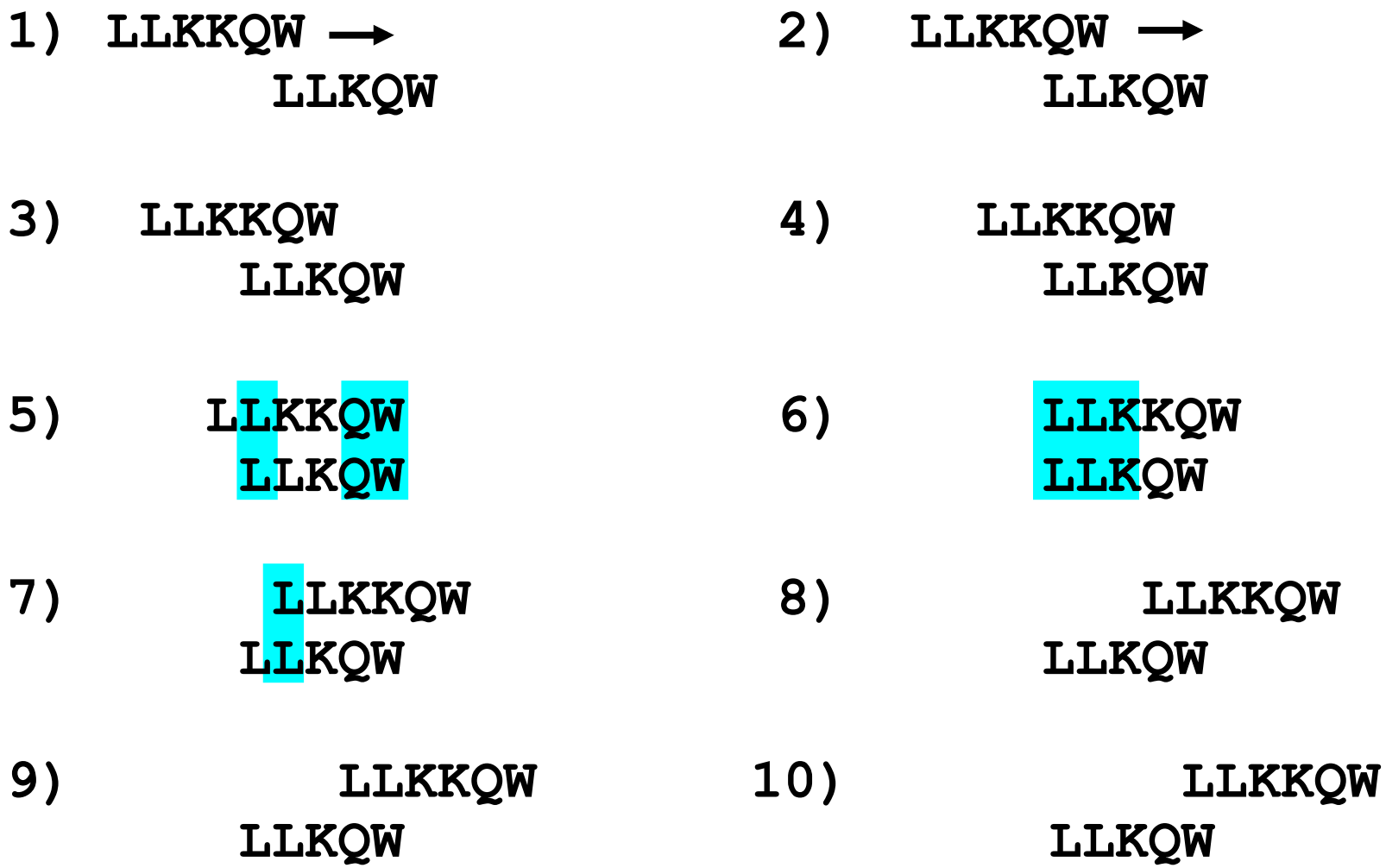
How do we discard the irrelevant alignments?



The heuristic algorithms (BLAST, FASTA) can filter most of the irrelevant alignments.

2 sequences of length *n* and *m* with a 'sliding' algorithm would require *n* x *m* comparisons between positions: problem O(*nm*) ~ O(n^2)* (quadratic size)

Example ***6*** x ***5***:



* If *n* and *m* have the same order of magnitude

What if we allow gaps?

In a sequence long ***n*** one can insert gaps in ***n***-1 positions.

Just allowing 1-res gaps “-” “***n***” different sequences are obtained

Example ***n*** = 6 :

1) **LLKKQW**	2) **L-LKKQW**
3) **LL-KKQW**	4) **LLK-KQW**
5) **LLKK-QW**	6) **LLKKQ-W**

What if we allow gaps?

In a sequence long ***n*** one can insert gaps in ***n***-1 positions.

Just allowing 1-res gaps "-" "***n***" different sequences are obtained

Example ***n*** = 6 :

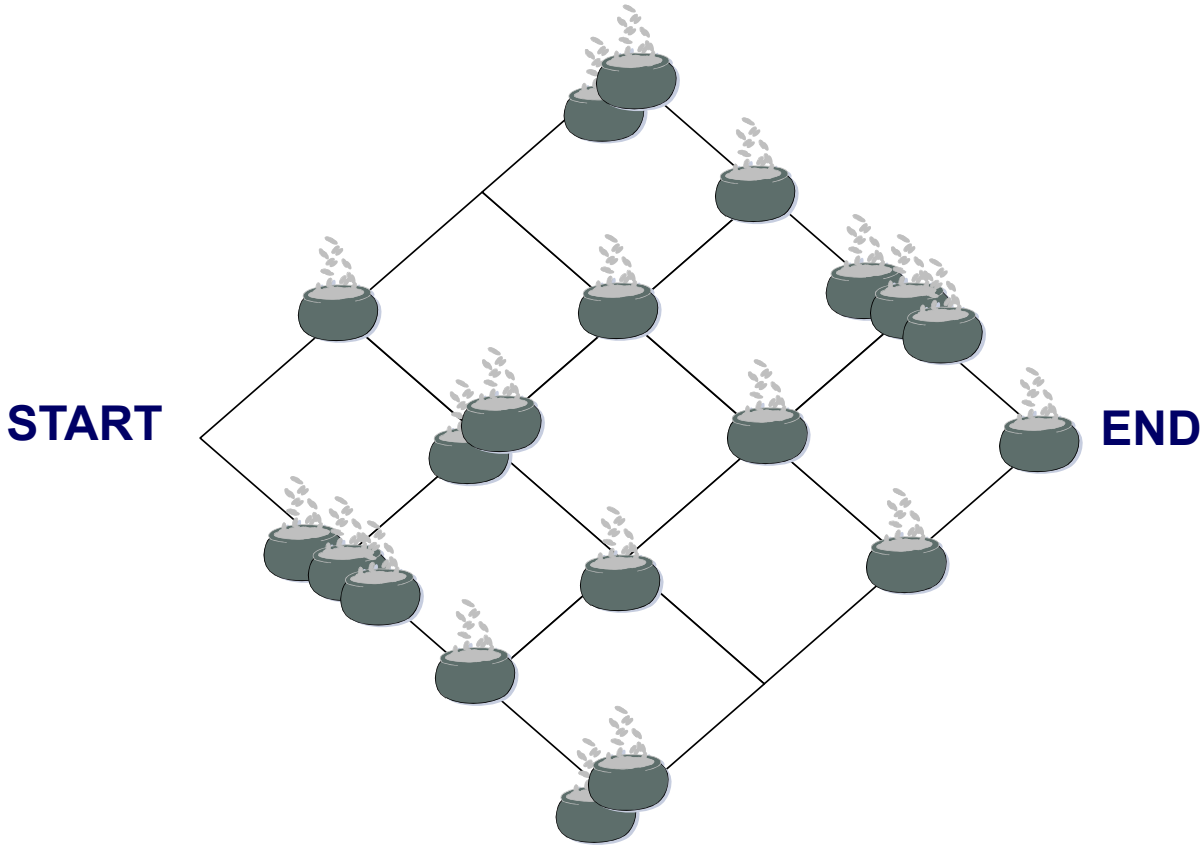
1) **LLKKQW**	2) **L-LKKQW**
3) **LL-KKQW**	4) **LLK-KQW**
5) **LLKK-QW**	6) **LLKKQ-W**

In case we allow a larger number of gaps (besides the 1-res) the number of possible sequences increases exponentially and so does the problem size

Exact alignment algorithms such as the Needlman-Wunch and Smith-Waterman are examples of Dynamic Programming

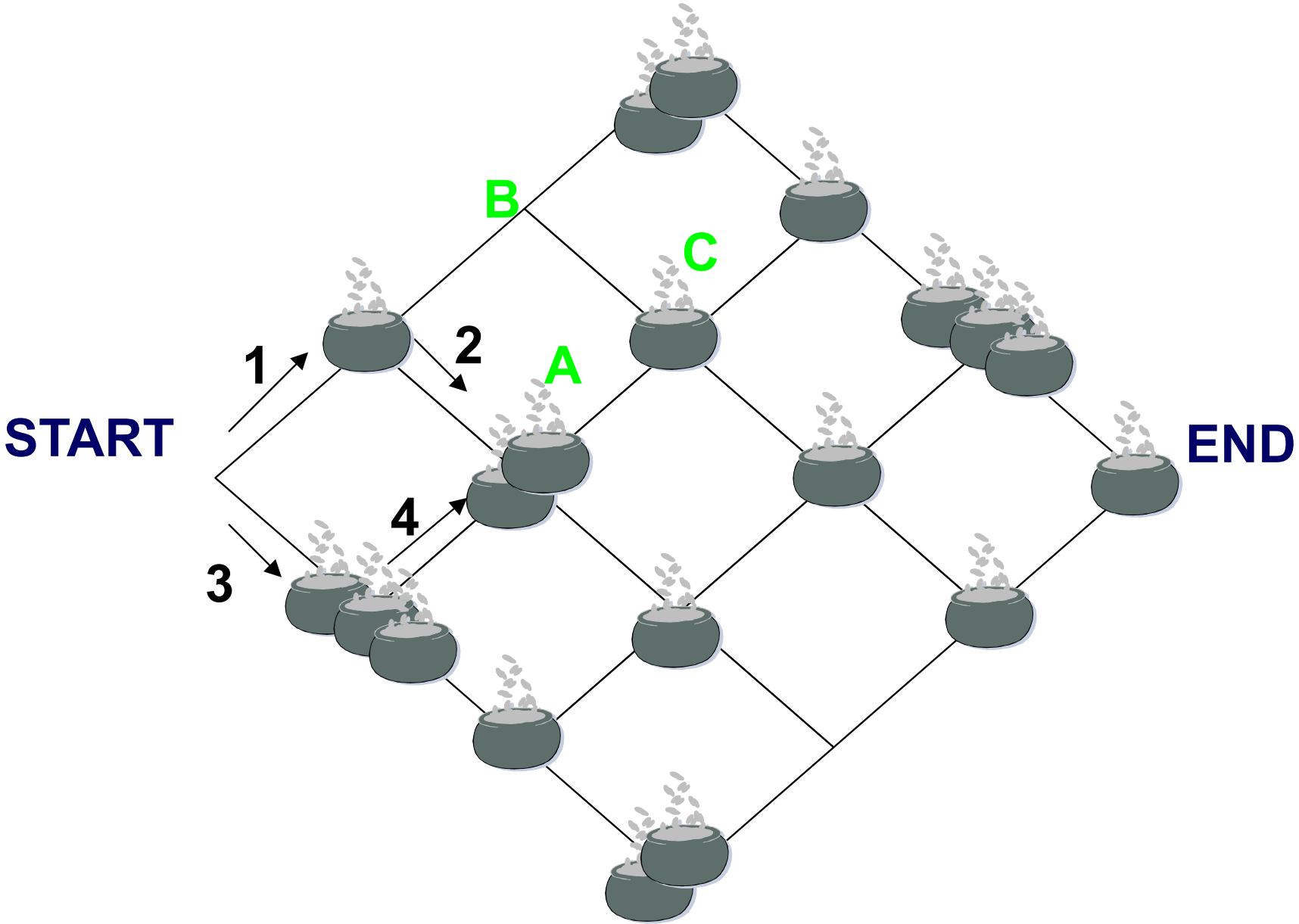
Breaking down the problem into simpler sub-problems, then recursively finding the optimal solutions to the sub-problems

Pots of gold game: rules

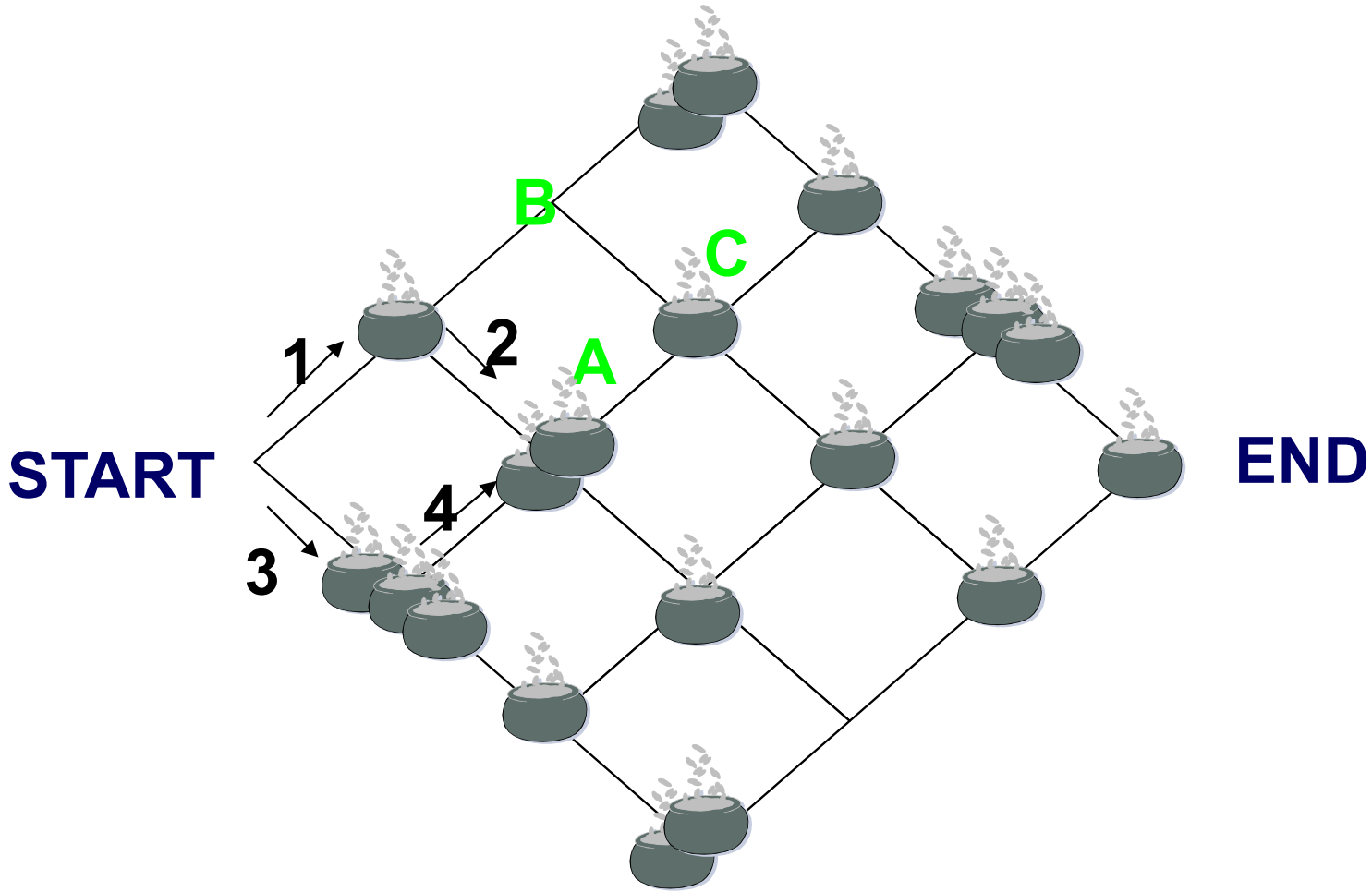


Going from START to END without passing twice through the same point and without moving backwards, while collecting the max number of pots of gold

Pots of gold game

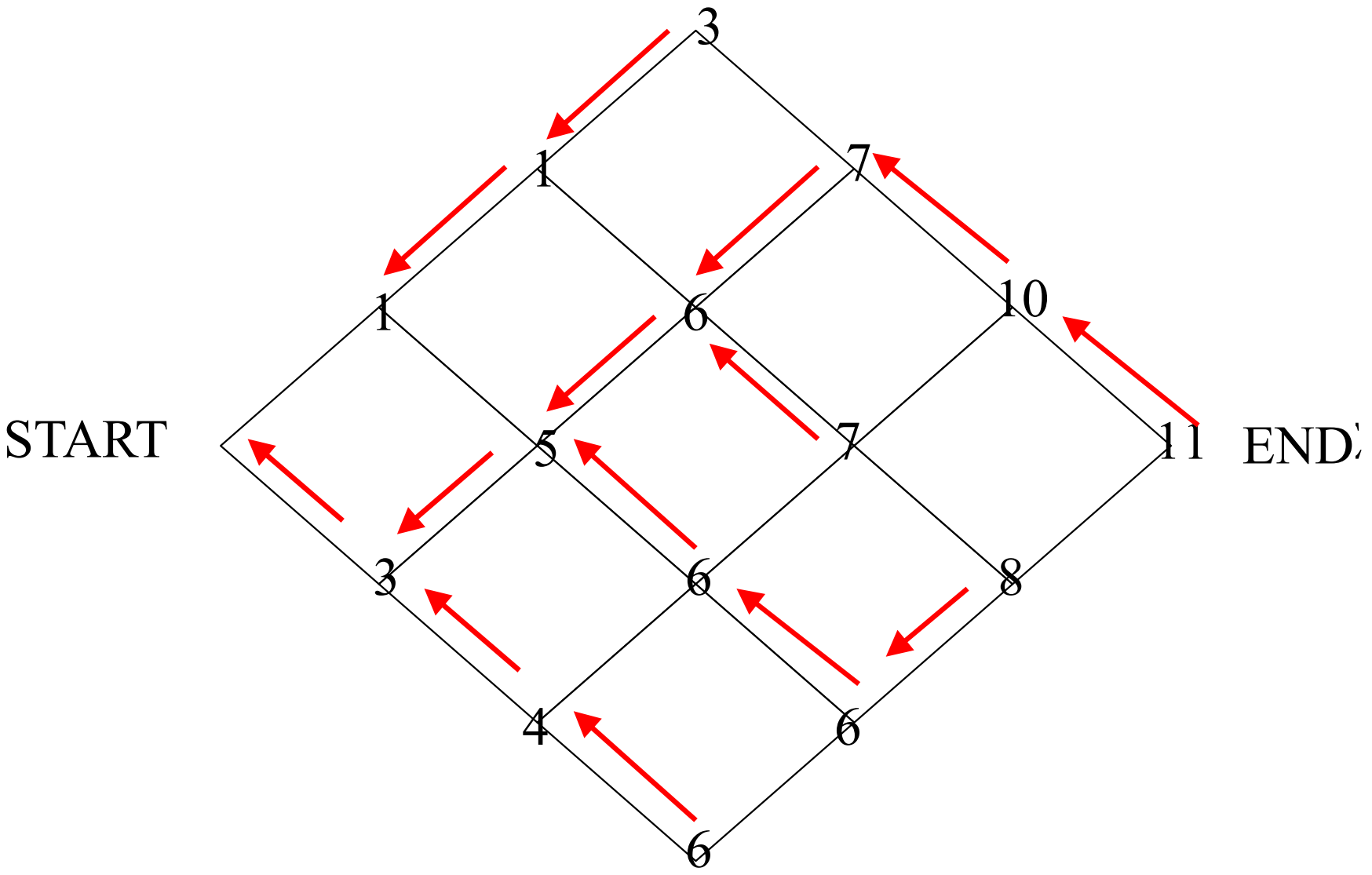


Pots of gold game

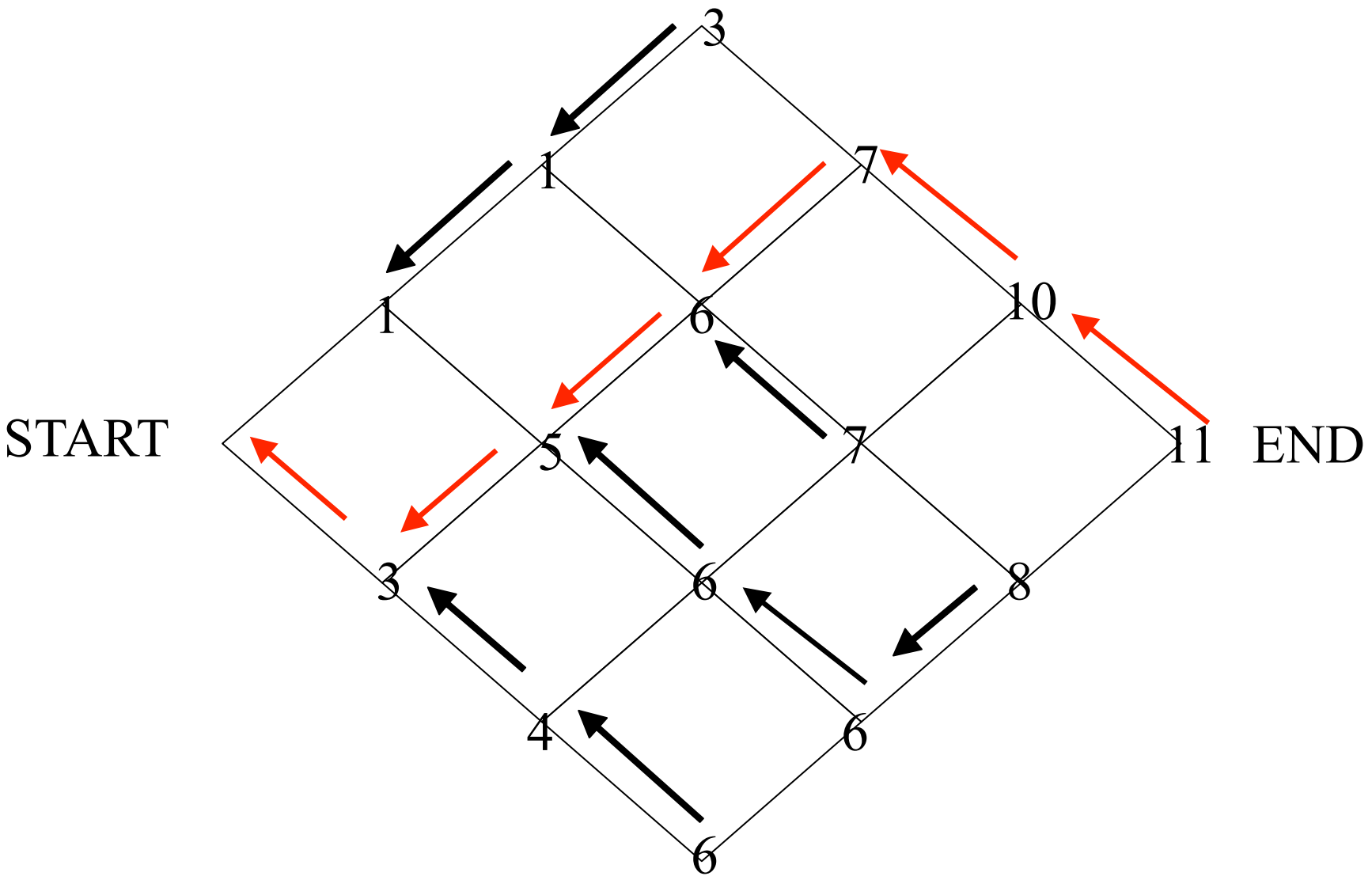


Choice of the best path from A to END does not depend on the path taking me from START to A

Pots of gold game: the optimal path (solution)



Pots of gold game: the optimal path (solution)



Differently from the pots of gold game, similarity matrices also feature negative values

An alignment algorithm will tend to skip them by inserting too many insertions and deletions (INDELs, *GAPs*)

It is necessary to associate a penalty to the introduction of GAPs (both *opening*, and *extension* ones) in the alignment

e.g. How we align the following sequences:

HEAGAWGHEE

PAWHEAE

?

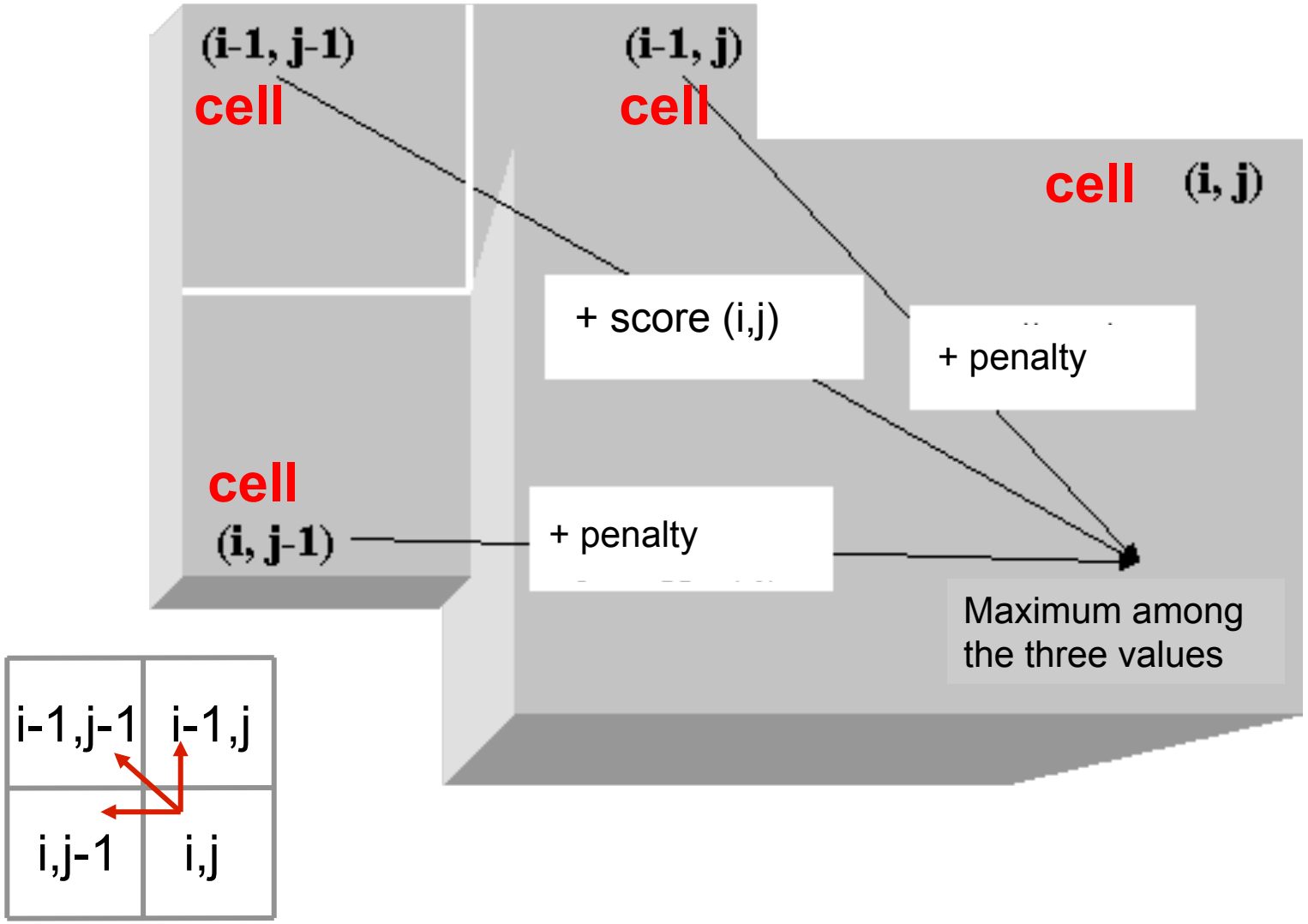
An example of alignment algorithms: cumulative matrix

HEAGAWGHEE *vs* PAWHEAE

Step1: building the site-specific matrix (values from BLOSUM45)

	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	5	0	5	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0	0
E	0	6	-1	-3	-1	-3	-3	0	6	6
A	-2	-1	5	0	5	-3	0	-2	-1	-1
E	0	6	-1	-3	-1	-3	-3	0	6	6

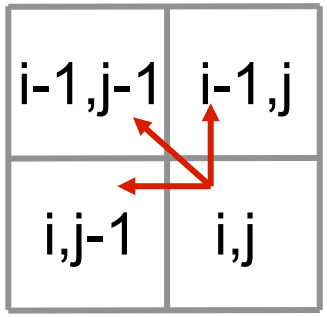
Alignment algorithm: cumulative matrix

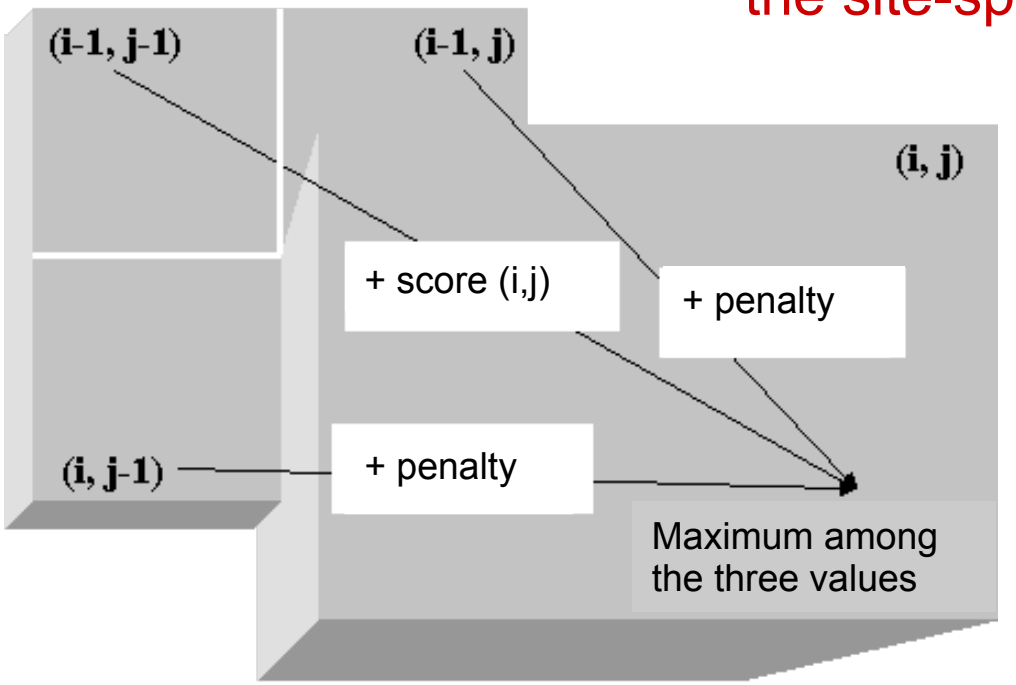


Step2: building the *cumulative* matrix, where each element represents the optimal score that can be obtained from start to that point

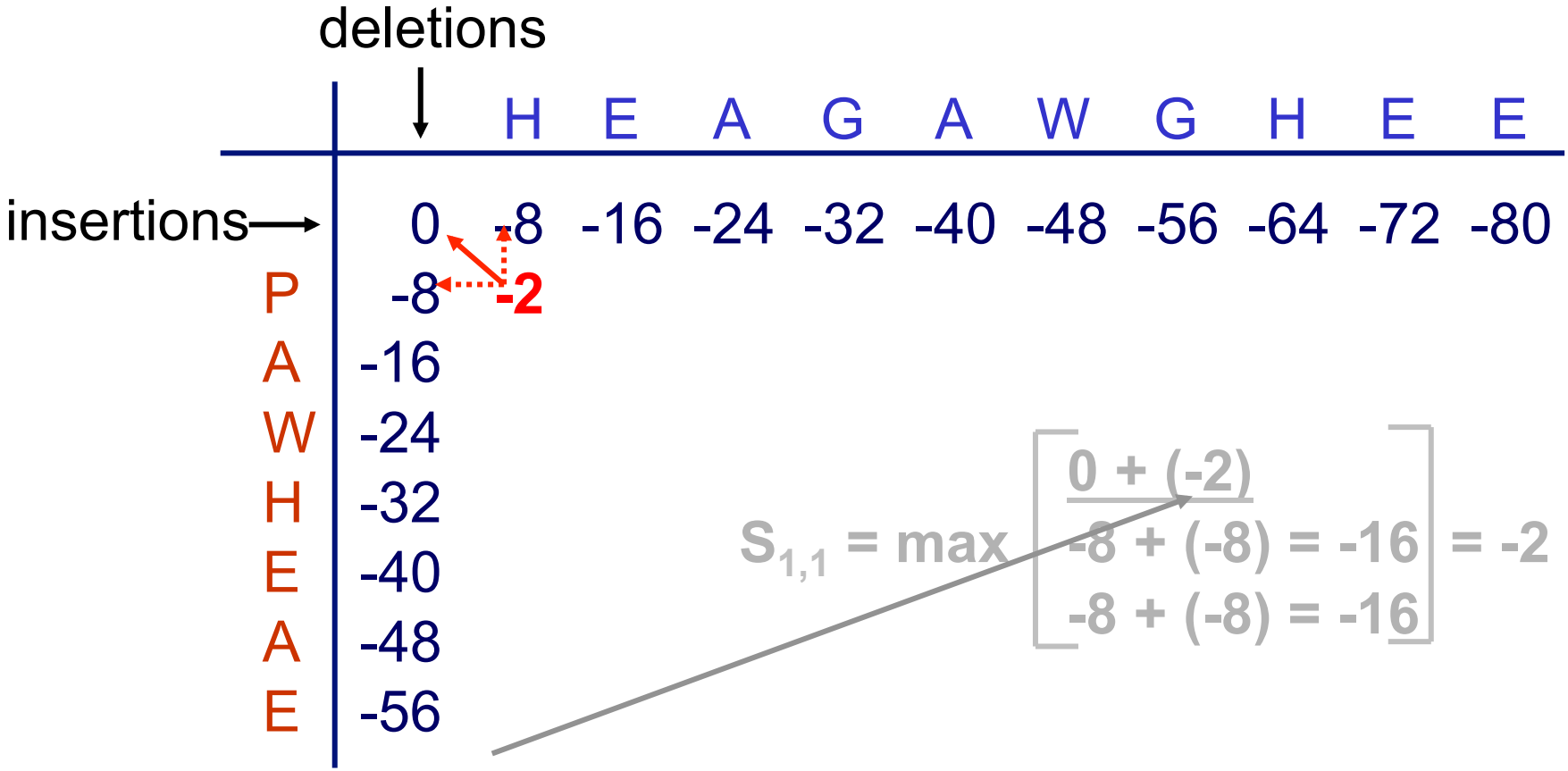
$$S_{i,j} = \max \begin{bmatrix} S_{i-1,j-1} + \sigma_{ij} \\ S_{i-1,j} + g \\ S_{i,j-1} + g \end{bmatrix}$$

where **g** represents the penalty for INDELs (gaps) and σ_{ij} is the score of the corresponding cell in the site-specific matrix





Cumulative matrix

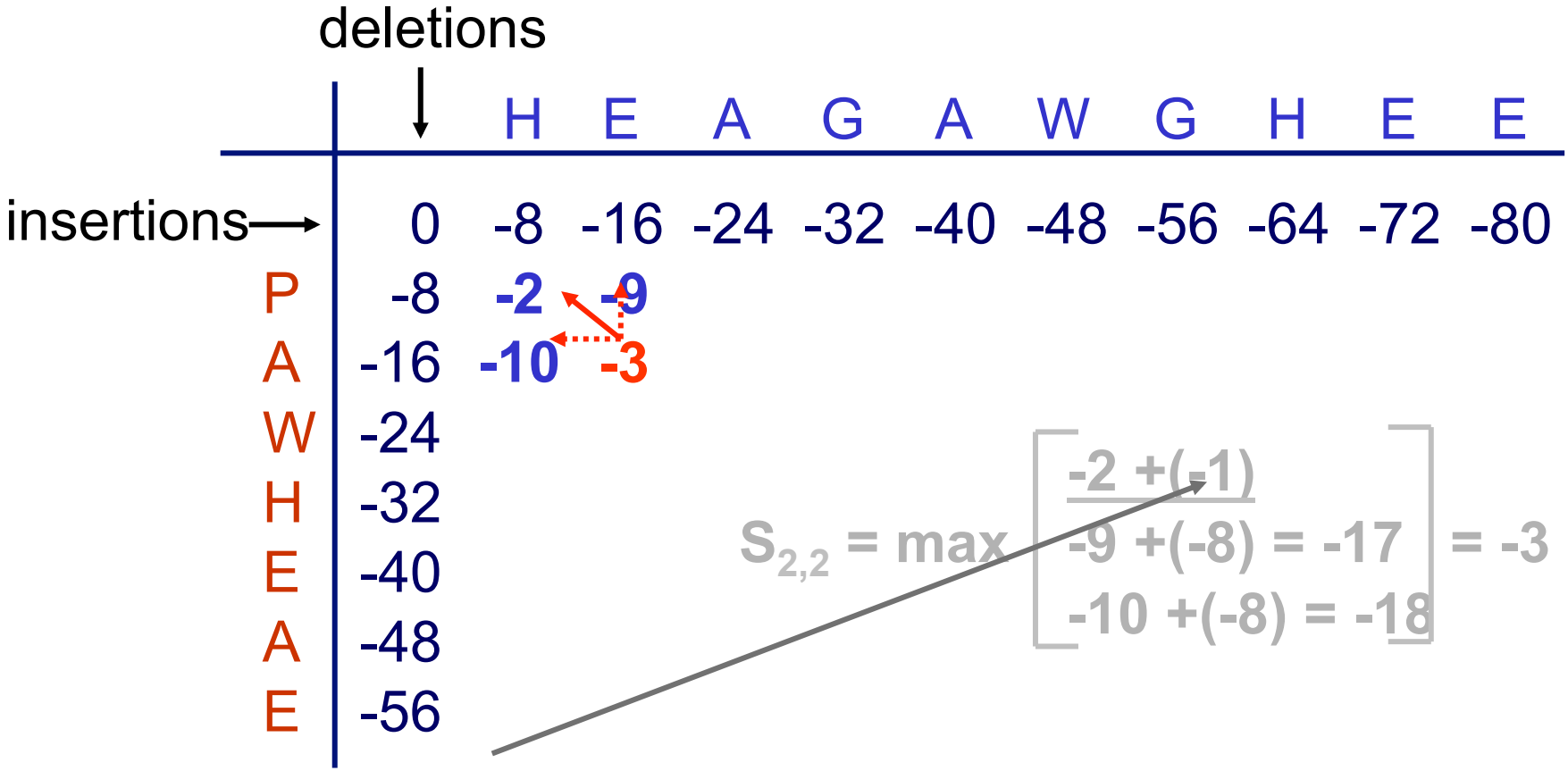


	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	5	0	5	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0	0
E	0	6	-1	-3	-1	-3	-3	0	6	6
A	-2	-1	5	0	5	-3	0	-2	-1	-1
E	0	6	-1	-3	-1	-3	-3	0	6	6

GAP penalty = -8

Original site-specific matrix

Cumulative matrix



	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	5	0	5	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0	0
E	0	6	-1	-3	-1	-3	-3	0	6	6
A	-2	-1	5	0	5	-3	0	-2	-1	-1
E	0	6	-1	-3	-1	-3	-3	0	6	6

GAP penalty = -8

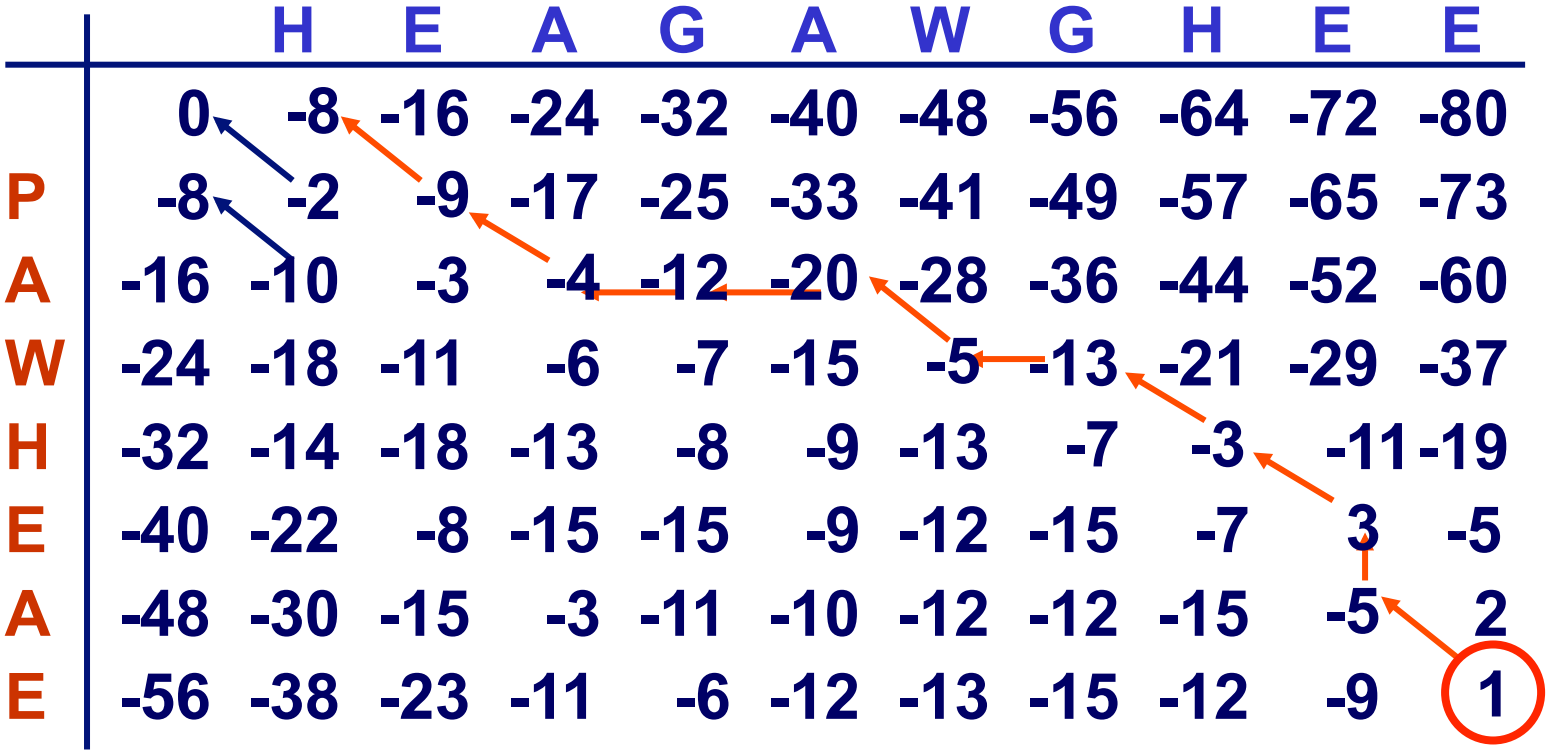
Original site-specific matrix

Cumulative matrix

		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-41	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-15	-15	-9	-12	-15	-7	3	-5
A	-48	-30	-15	-3	-11	-10	-12	-12	-15	-5	2
E	-56	-38	-23	-11	-6	-12	-13	-15	-12	-9	1

	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	5	0	5	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0	0
E	0	6	-1	-3	-1	-3	-3	0	6	6
A	-2	-1	5	0	5	-3	0	-2	-1	-1
E	0	6	-1	-3	-1	-3	-3	0	6	6

Original site-specific matrix



maximum achievable score

HEAGAWGHE-E

-PA--W-HEAE

Step3: backwards path through cells that allowed to obtain the best alignment scores

		H	E	A	G	A	W	G	H	E	E
	0	**-8**	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	**-9**	-17	-26	-33	-44	-50	-58	-65	-73
A	-16	-10	-3	**-4**	**-12**	**-15**	-26	-34	-44	-53	-62
W	-24	-19	-13	-6	-7	-15	**0**	-11	-22	-33	-44
H	-32	-14	-19	-15	-8	-9	-11	**-2**	**0**	-8	-16
E	-40	-22	-8	-17	-18	-9	-12	-13	-2	**6**	4
A	-48	-32	-17	-3	-11	-12	-12	-12	-12	**-3**	5
E	-56	-40	-19	-12	-6	-12	-15	-15	-12	-5	**3**

An example of alignment algorithms: cumulative matrices

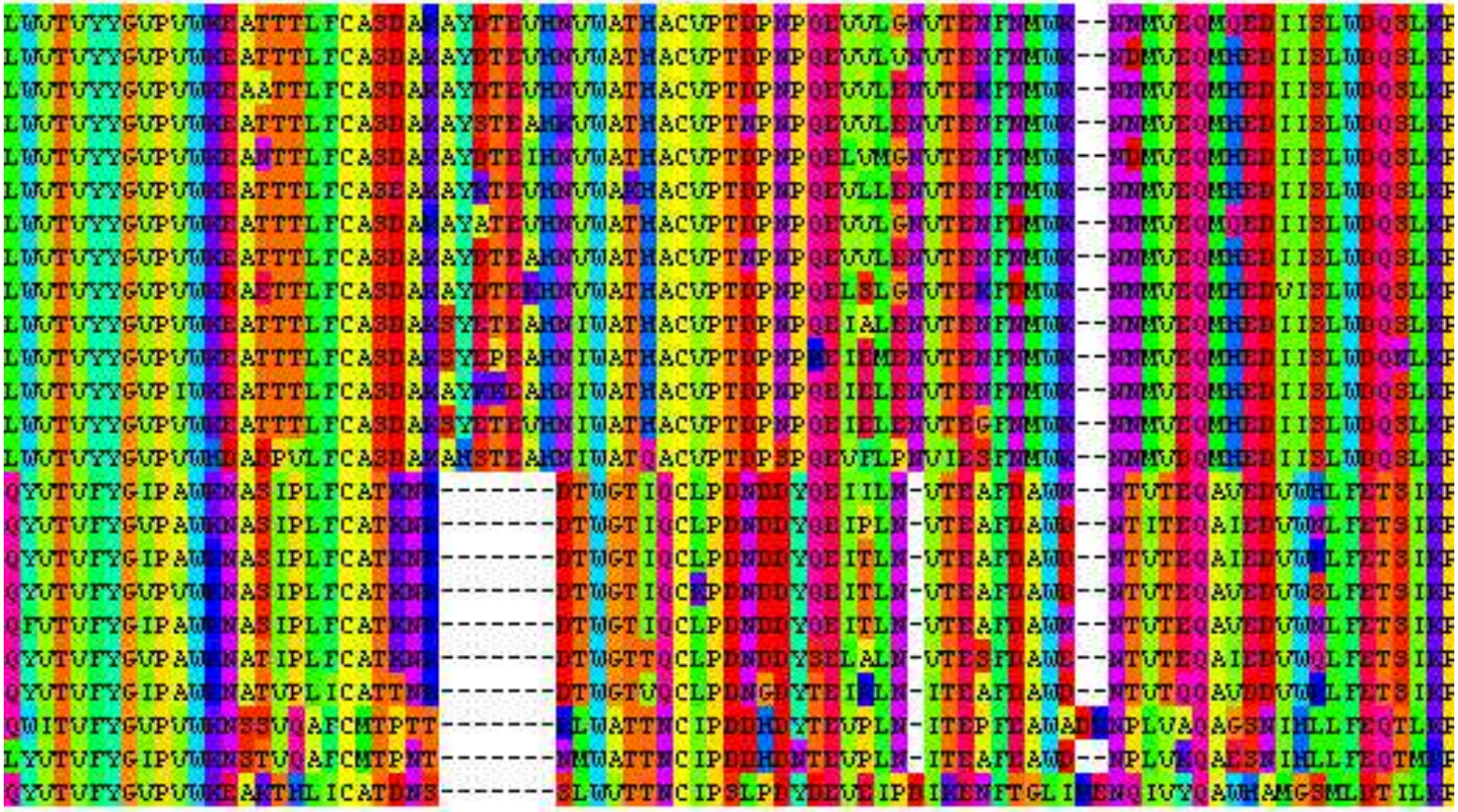
HEAGAWGHEE *vs* PAWHEAE

Step1: building a site-specific matrix (values from a PAM or BLOSUM matrix)

Step2: building a *cumulative* matrix, where each element represents the maximum score achievable to go from start to that point

Step3: backwards path through the cells which allowed to obtain the best scores = optimal alignment

Multiple alignment



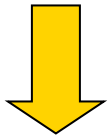
Few definitions…

Pair-wise sequence alignment =
Alignment of TWO similar sequences

Multiple sequence allignment (MSA) =
Alignment of MANY similar sequences,
generally coming from a search in databases

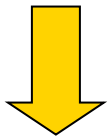
Exact algorithms for multiple alignments

The necessary number of computational steps is in the order L^N, where L is the length and N the number of sequences to be aligned



e.g. for 4 sequences 100 aa/nt long

The number of computational steps would be 100^4 = 100 millions !



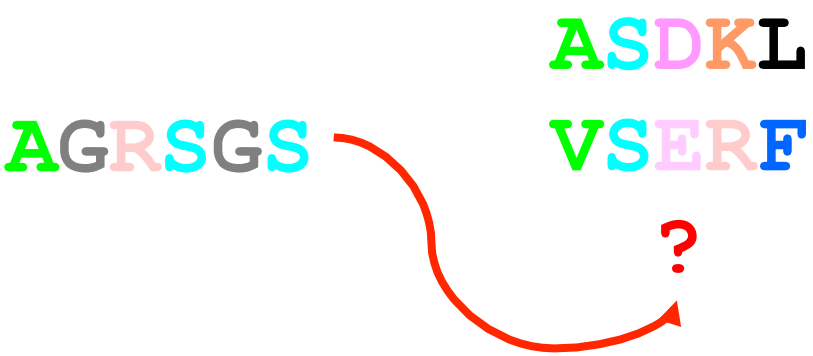
not viable

Approximate solution

We perform a pairwise alignment(s) then align additional sequences to it(them)

and so on…

Approximate solution

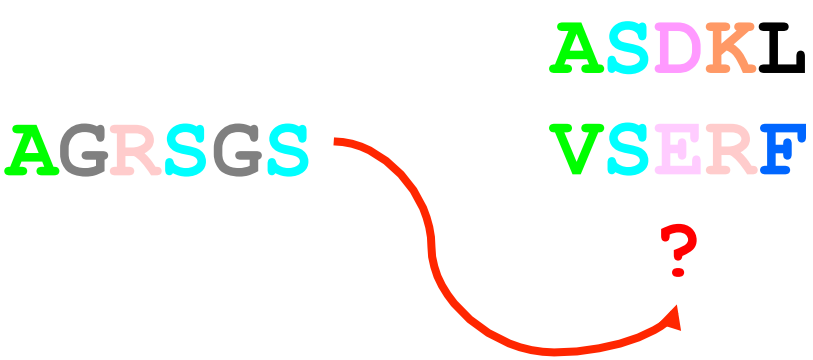


A profile of the alignment can be built in the form of a PSSM (position-specific scoring matrix) but also by HMM

	A V	S S	D E	K R	L F
A	½(PAM(A,A)+PAM(A,V))=½(2+0)	½(1+1)	½(0+0)	½(-1-2)	½(-2-4)
G	½(1-1)	½(1+1)	½(1+0)	½(-2-3)	½(-4-5)
R	½(-2-2)	½(0+0)	½(-1-1)	½(3+6)	½(-3-4)
S	½(1+0)	½(3+3)	½(0+0)	½(0+0)	½(-3-3)
G	½(1-1)	½(1+1)	½(1+0)	½(-2-3)	½(-4-5)
S	½(1+0)	½(3+3)	½(0+0)	½(0+0)	½(-3-3)

Example of PSSM

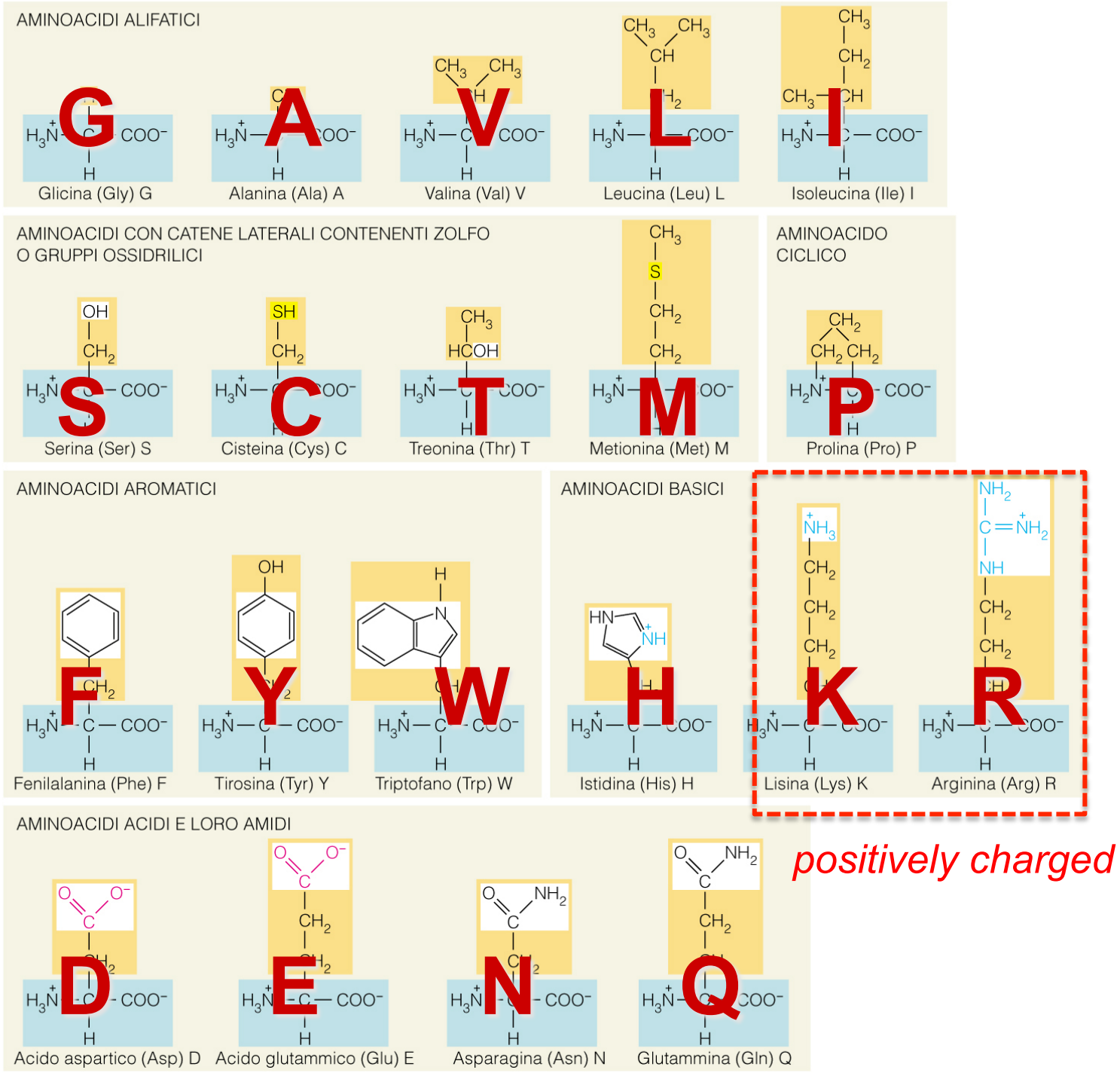
Approximate solution



The easiest way to go is building a PSSM by performing an arithmetic average of the scores for the alignment of ress of the 3rd to those of the aligned sequences

	A V	S S	D E	K R	L F
A	½(PAM(A,A)+PAM(A,V))=½(2+0)	½(1+1)	½(0+0)	½(-1-2)	½(-2-4)
G	½(1-1)	½(1+1)	½(1+0)	½(-2-3)	½(-4-5)
R	½(-2-2)	½(0+0)	½(-1-1)	½(3+6)	½(-3-4)
S	½(1+0)	½(3+3)	½(0+0)	½(0+0)	½(-3-3)
G	½(1-1)	½(1+1)	½(1+0)	½(-2-3)	½(-4-5)
S	½(1+0)	½(3+3)	½(0+0)	½(0+0)	½(-3-3)

Example of PSSM


AMINOACIDI ALIFATICI
Glicina (Gly) G
Alanina (Ala) A
Valina (Val) V
Leucina (Leu) L
Isoleucina (Ile) I
AMINOACIDI CON CATENE LATERALI CONTENENTI ZOLFO O GRUPPI OSSIDRILICI
Serina (Ser) S
Cisteina (Cys) C
Treonina (Thr) T
Metionina (Met) M
AMINOACIDO CICLICO
Prolina (Pro) P
AMINOACIDI AROMATICI
Fenilalanina (Phe) F
Tirosina (Tyr) Y
Triptofano (Trp) W
AMINOACIDI BASICI
Istidina (His) H
Lisina (Lys) K
Arginina (Arg) R
positively charged
AMINOACIDI ACIDI E LORO AMIDI
Acido aspartico (Asp) D
Acido glutammico (Glu) E
Asparagina (Asn) N
Glutammina (Gln) Q

	A	S	D	K	L
A	2	1	0	-1	-2
G	1	1	1	-2	-4
R	-2	0	-1	3	-3
S	1	3	0	0	-3
G	1	1	1	-2	-4
S	1	3	0	0	-3

	A V	S S	D E	K R	L F
A	1	1	0	-1.5	-3
G	0	1	0.5	-2.5	-4.5
R	-2	0	-1	4.5	-3.5
S	0.5	3	0	0	-3
G	0	1	0.5	-2.5	-4.5
S	0.5	3	0	0	-3

ASDKL

VSERF

A-GRSGS

A classical approach: CLUSTALW

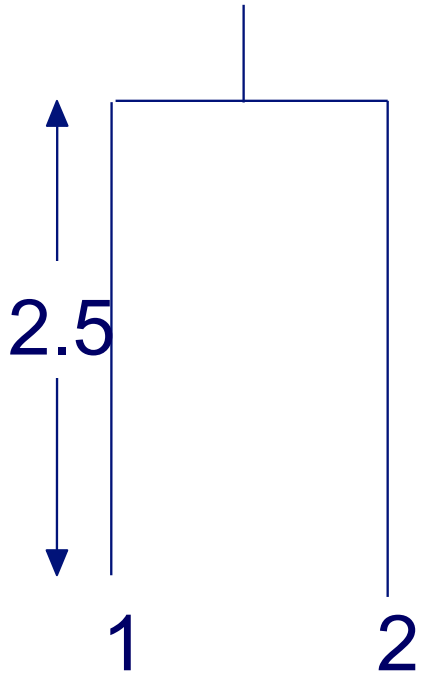
- Programme for MSAs based on a **hierarchic approach**
- Step1: pairwise alignment for all the input sequences (for N seq.: N(N-1)/2 pairwise alignments)
- Step2: building a guide tree, i.e. a hierarchy of sequences in order of their similarity (cluster analysis)
- Step3: building the multiple alignment based on the guide tree by first aligning the most similar pairs, then aligning the other sequences with those pairs until all have been aligned

Measured distance = % of different amino acids

	Seq1	Seq2	Seq3	Seq4
Seq1	0	5	11	14
Seq2		0	9	10
Seq3			0	7
Seq4				0

The measure of dissimilarity between sequences represents their evolutionary distance and can be obtained in several ways

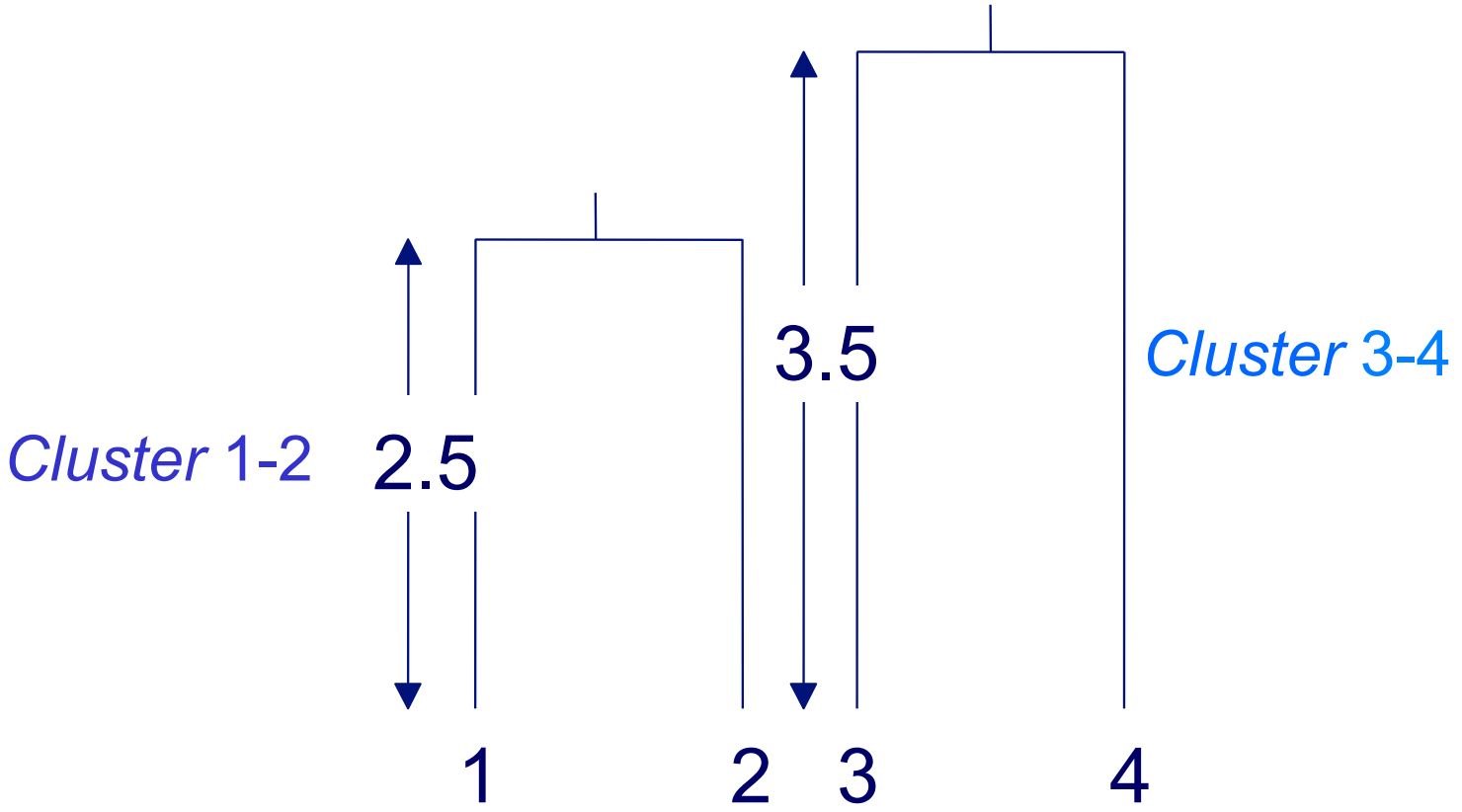
The % of different amino acids is one possible way



Cluster 1-2

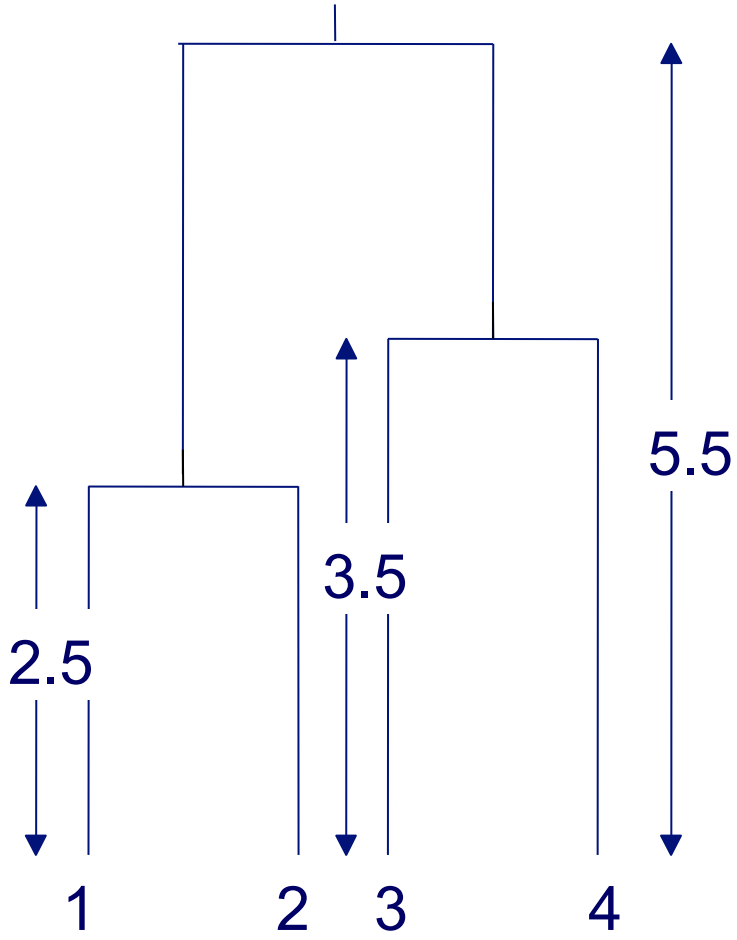
Measured distance = % of different amino acids

	Cls1-2	Seq3	Seq4
Cls1-2	0	½[d(1,3)+d(2,3)] = 10	½[d(1,4)+d(2,4)] = 12
Seq3		0	7
Seq4			0



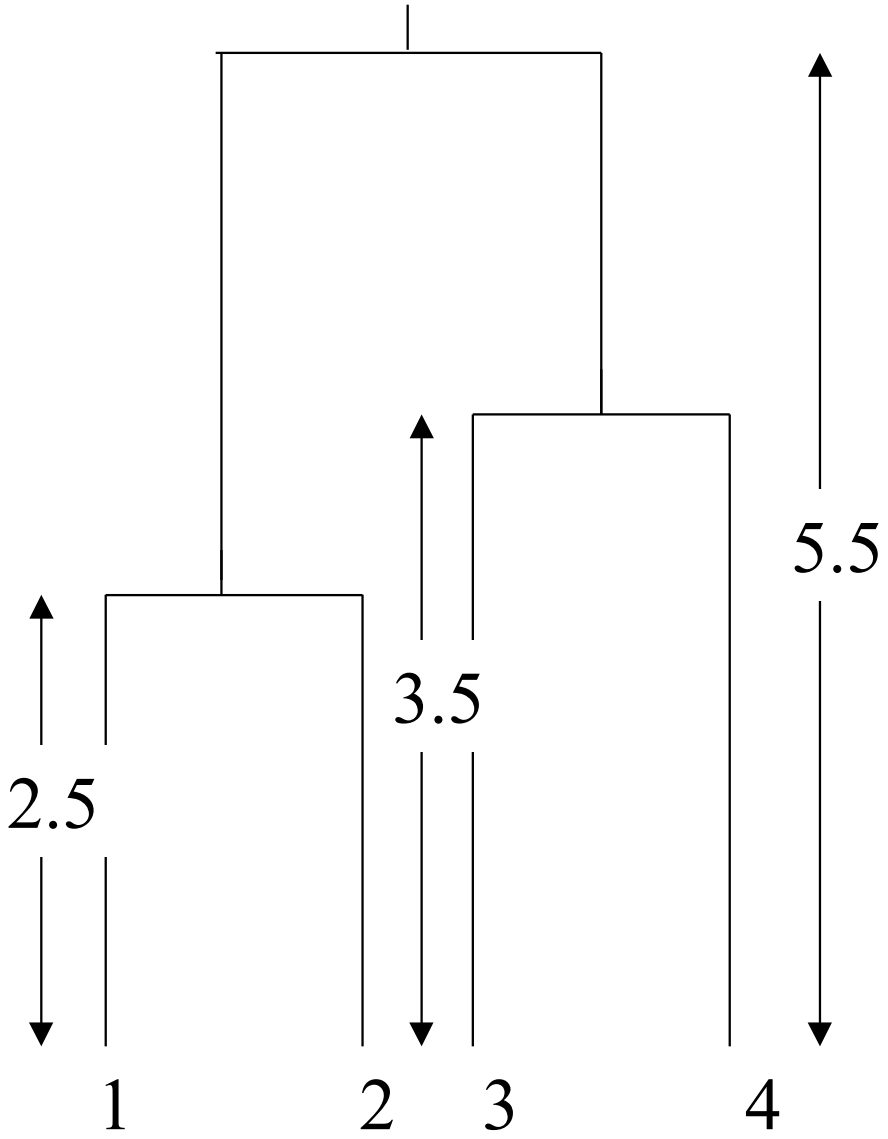
Measured distance = % of different amino acids

	Cls1-2	Cls 3-4
Cls1-2	0	½[d(Cls 1-2),3] + ½[d(Cls 1-2),4] = 11
Cls 3-4		0



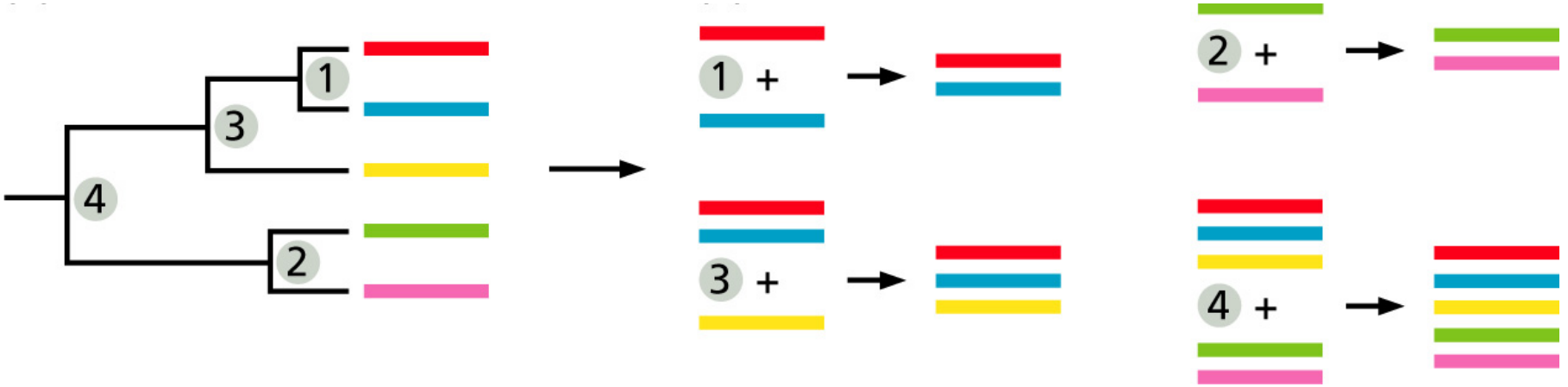
Example of guide tree

The guide tree is used to guide the order of constructing the multiple alignment



Another example of guide tree

How a guide tree, built on the basis of pair-wise alignments between all the sequences, guides the order of constructing the multiple alignment



In principle, the pair-wise alignments can also be approximate, for instance based on the presence of k*-mers (stretches of* k *residues) in common for two sequences*

MSAs: current approaches

- ClustalW has been retired and substituted as a web service by Clustal Omega
- Clustal Omega is a new MSA tool that uses seeded guide trees and HMM profile-profile techniques to generate alignments

Hidden Markov Models
(HMMs)

A HMM is a **machine learning method**: it learns from known cases, assigns probabilities to events based on observations and makes predictions

HMMs can be designed for many tasks, in Bioinformatics and beyond

Markov Model means that there is a statistical **Markov chain**, i.e. each state at the step *n* depends on the state at the step *n*-x (if x=1, the Markov chain is of first order)

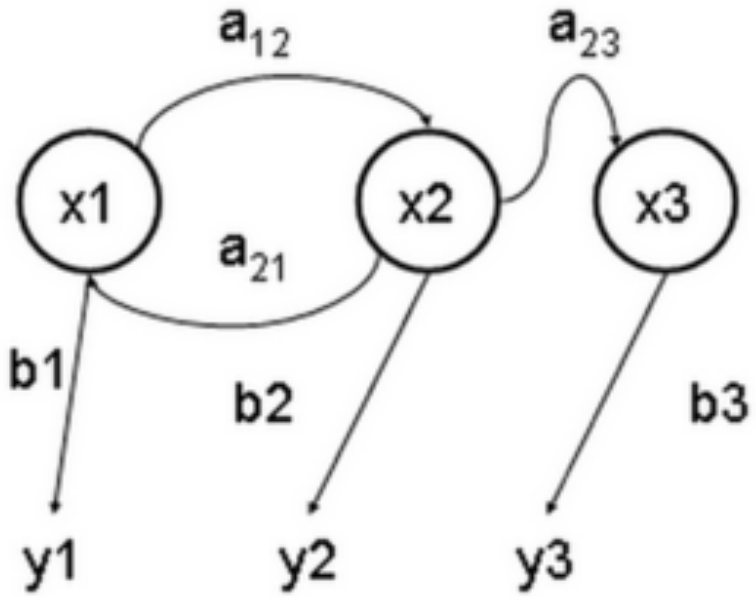
Hidden means that its **states are not observable**

HMM structure

A HMM is characterized by:

Number of states (X), number of symbols (Y), distribution of transition probabilities between states (A), distribution of emission probabilities of symbols (B), initial state distribution (π).

States can be "silent", in case they do not emit symbols



Hidden Markov Models
(HMMs)

Hidden Markov models are made of unobservable (hidden) states

Each state emits symbols from a fixed alphabet, e.g. ACGT, according to specific emission (or output) probabilities

Different (hidden) states are connected by precise transition probabilities

The sequence of states is a Markov chain: the choice of next element (state) depends on the actual one (1st order chain)

While states are “hidden”, symbols (e.g. ACGT) are “observable”

Observations are probability functions of the “hidden” states

Hidden Markov Models
(HMMs)

Hidden Markov models are made of unobservable (hidden) states

Each state emits symbols from a fixed alphabet, e.g. ACGT, according to specific emission (or output) probabilities

Different (hidden) states are connected by precise transition probabilities

The sequence of states is a Markov chain: the choice of next element (state) depends on the actual one

While states are “hidden”, symbols (e.g. ACGT) are “observable”

We can think of a HMM as a generator of sequences with defined probabilities

Hidden Markov Models (HMMs)

Hidden Markov models are made of unobservable (hidden) states

Each state emits symbols from a fixed alphabet, e.g. ACGT, according to specific emission (or output) probabilities

Different (hidden) states are connected by precise transition probabilities

The sequence of states is a Markov chain: the choice of next element (state) depends on the actual one

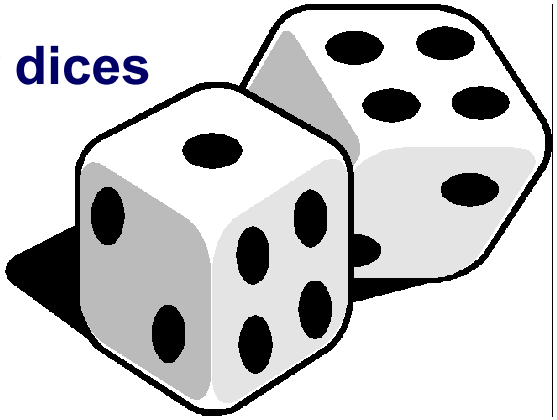
While states are “hidden”, symbols (e.g. ACGT) are “observable”

Basically we use a sequence of observations to estimate the sequence of hidden states

Hidden-Markov Models (HMMs)

In this example of HMM the hidden states are the used dices (regular or modified) and the outcomes of rolls are the symbols (observable)

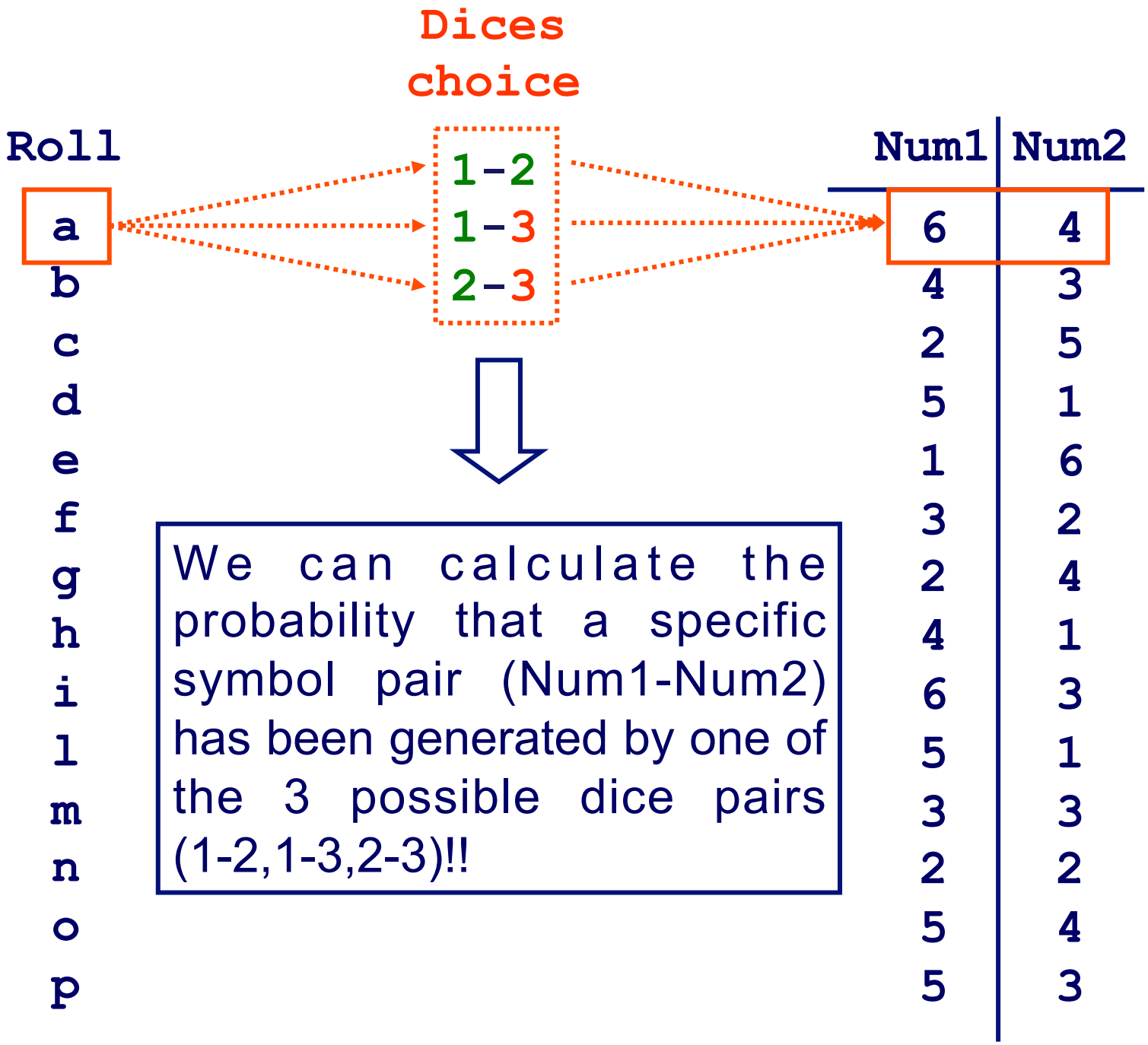
Regular dices



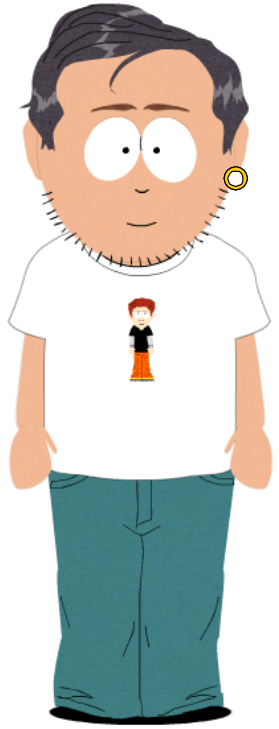
Rigged (modified) dice: Same external features of the regular dices pair

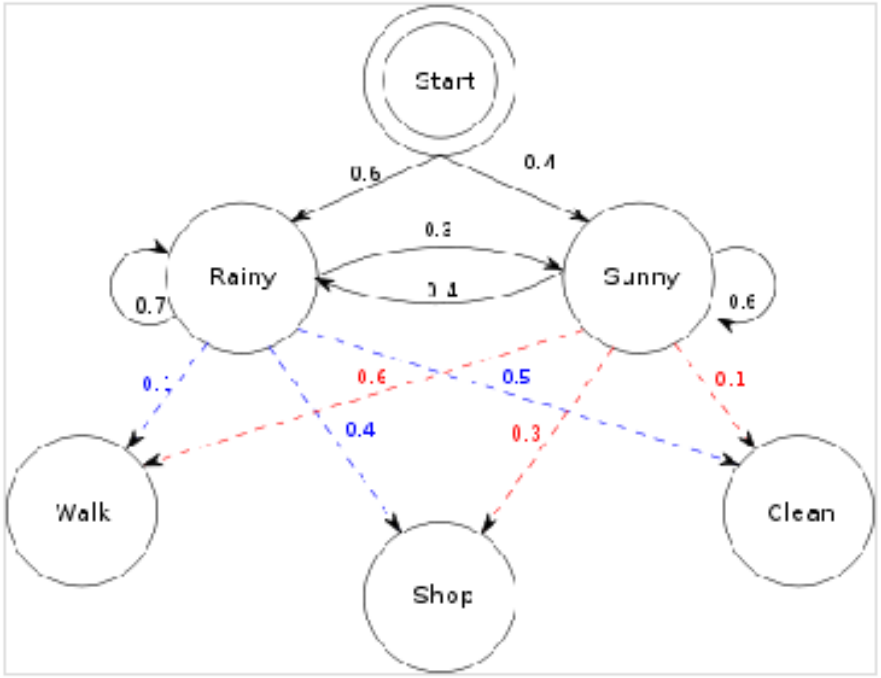
Roll	**abcdefghilmnop**
Num1	**64251324653255**
Num2	**43516241313243**

?? Given the above sequence of symbols, what pair of dices have we thrown ??


Dices choice
Roll
a
b
c
d
e
f
g
h
i
l
m
n
o
p
1-2
1-3
2-3
Num1
Num2
6 4
4 3
2 5
5 1
1 6
3 2
2 4
4 1
6 3
5 1
3 3
2 2
5 4
5 3
We can calculate the probability that a specific symbol pair (Num1-Num2) has been generated by one of the 3 possible dice pairs (1-2,1-3,2-3)!!

One more example of HMM





Python representation of HMM parameters

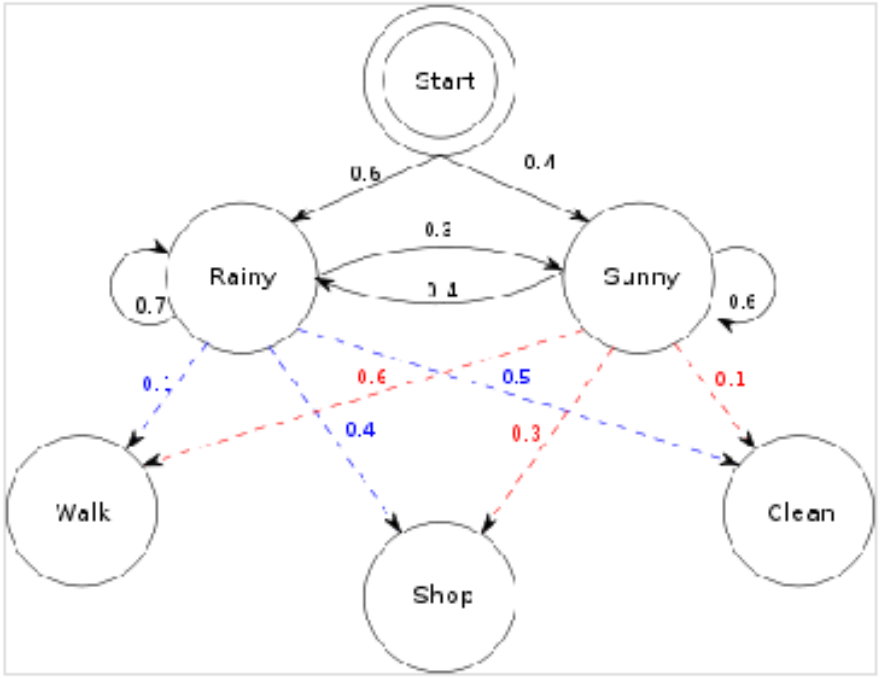
```
states = ('Rainy', 'Sunny')

observations = ('walk', 'shop', 'clean')

start_probability = {'Rainy': 0.6, 'Sunny': 0.4}

transition_probability = {
   'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3},
   'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
   }

emission_probability = {
   'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
   'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},
   }
```



Another example of HMM for a sequence alignment

The most likely sequences of observables emitted are:

PATH

PETH

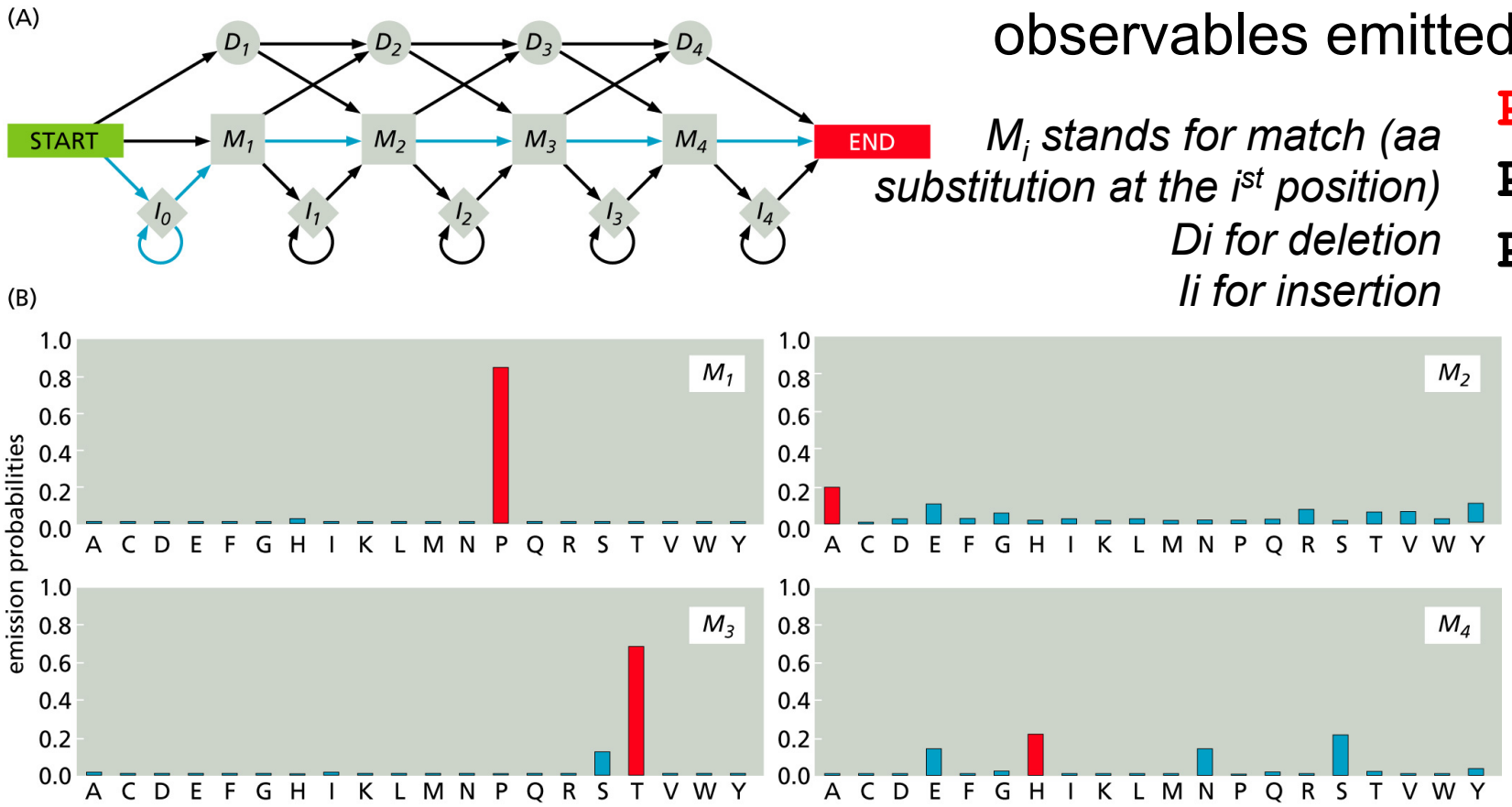
PETS

etc.

M_i stands for match (aa substitution at the i^{st} position)

Di for deletion

Ii for insertion



The hidden states here are: deletion, insertion and amino acid matches. Deletions are instances of “silent states”

Basic problems for HMMs

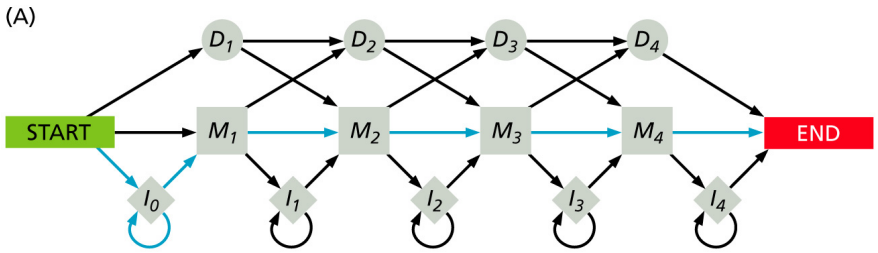
Given the structure of a HMM (X,Y,A,B,π)

Problem 1: how we calculate the probability of a sequence of observations (e.g. “LASD”) O = $O_1O_2O_3…O_n$? (forward-backward algorithm)

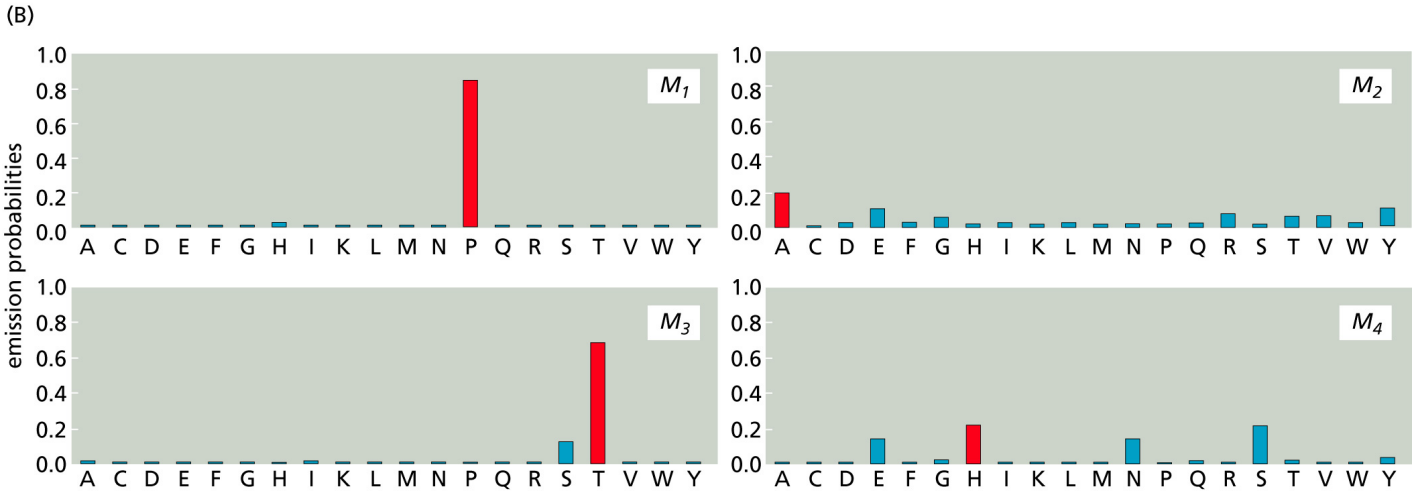
Problem 2: given a sequence of observations, how we choose an optimal sequence of states, which ‘explains’ the sequence of observations? (Viterbi algorithm).

Problem 3: how we adjust the parameters (transition probabilities) of the model to maximize the probability of a sequence of observations? (Baum-Welch algorithm) – this is how a HMM MSA is built

Another example of HMM for a sequence alignment



Given a HMM model, any given path through the model will emit a sequence with an associated probability



With the forward-backward algorithm, we can calculate the probability of having a specific sequence, e.g. **PETS** (problem 1 - *it will be the sum of all the paths emitting the sequence*)

With the Viterbi algorithm, we can choose the optimal sequence of states (most probable *path*), which 'explains' the sequence (problem 2); this is analogous to the best-scoring aln in dynamic programming

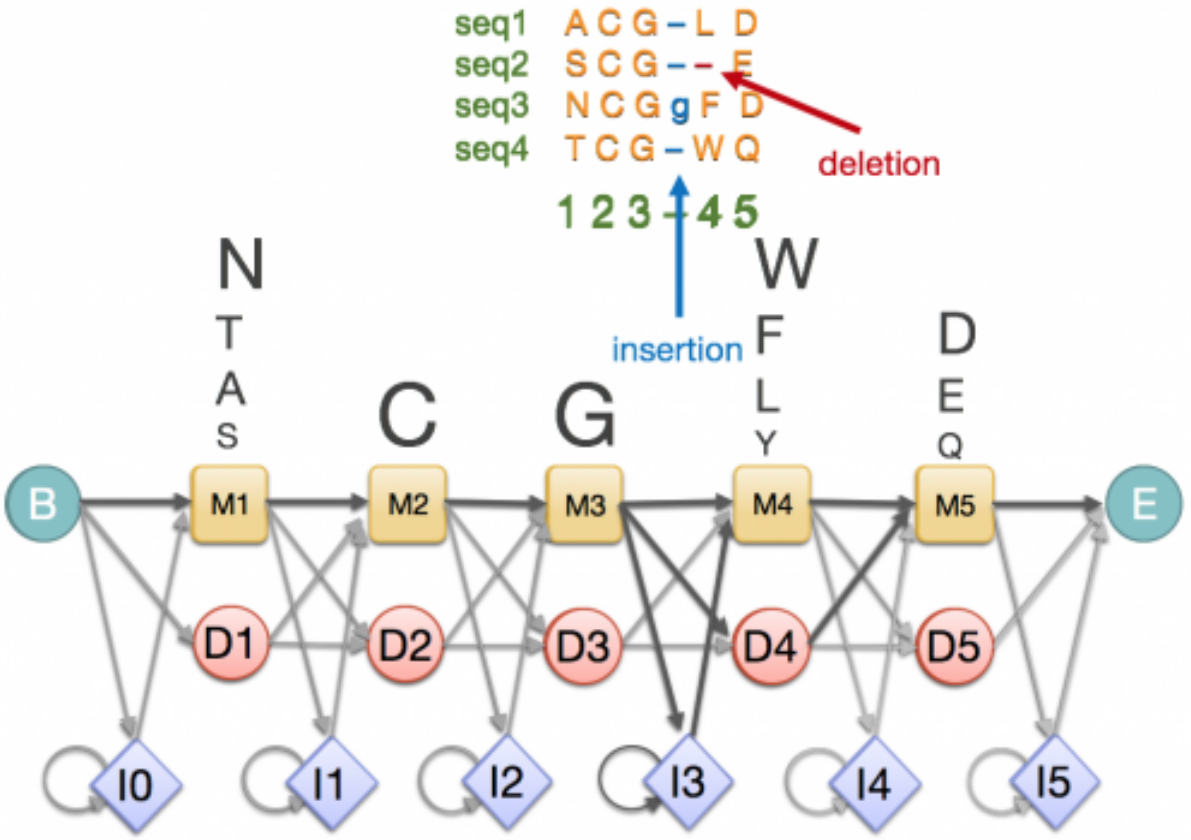
HMM profiles (or profile HMMs)

Start with a multiple sequence alignment

Insertions / deletions can be modelled

Occupancy and amino acid frequency at each position in the alignment are encoded

Profile created

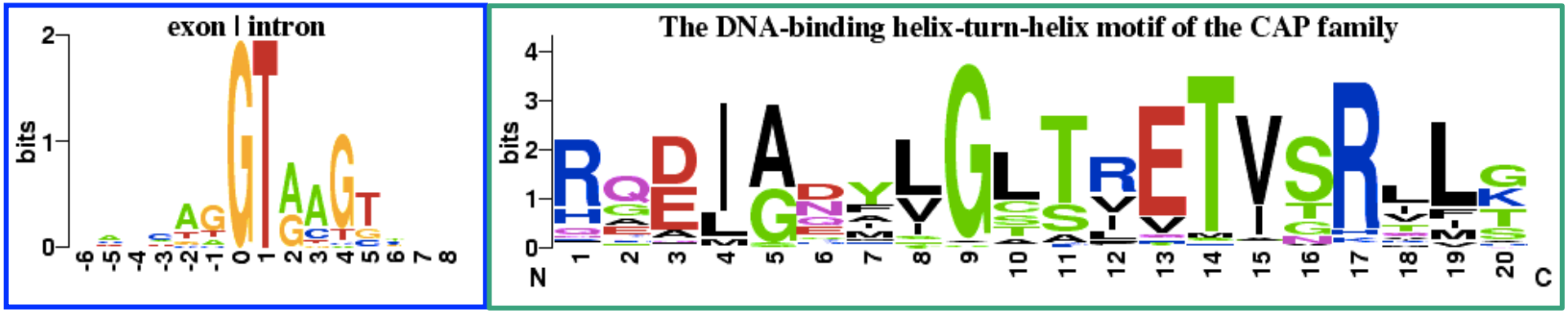


https://www.ebi.ac.uk/training/online/courses/pfam-creating-protein-families/what-are-profile-hidden-markov-models-hmms/

Sequence logos

Profiles of MSAs can be represented graphically in the form of sequence logos, easily showing the residue preference or conservation at particular positions, which point to a functional role

We have already encountered PSSMs (Position Specific Scoring Matrices), examples of scoring schemes of MSAs for searching for other similar sequences, represented as sequence logos



Examples from Web Logo

HMM profiles (or profile HMMs)

HMMs are commonly used to align a novel sequence to a HMM profile or to align HMM profiles one to each other

A HMM profile is a HMM model containing the information present in a multiple alignment

HMM profiles can be visualized using sequence logos to illustrate the emission probabilities for different residues types at each match state

They are more sophisticated versions of a PSSM, especially because they can treat INDELs in a position-dependent way

HMM profiles of protein families and subfamilies are reported in several databases (BLOCKS, Pfam etc.)

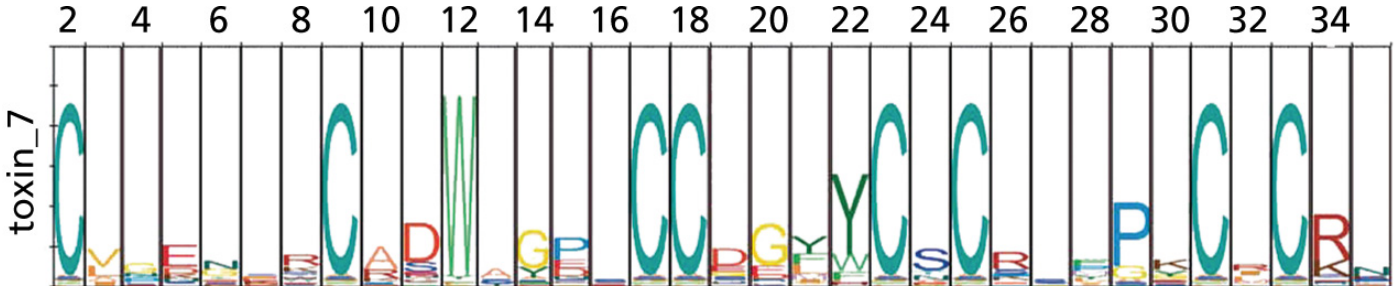
HMM profiles (or profile HMMs)

HMMs are commonly used to align a novel sequence to a HMM profile or to align HMM profiles one to each other

A HMM profile is a HMM model containing the information present in a multiple alignment

HMM profiles can be visualized using sequence logos to illustrate the emission probabilities for different residues types at each match state

They are more sophisticated versions of a PSSM, especially because they can treat INDELs in a position-dependent way



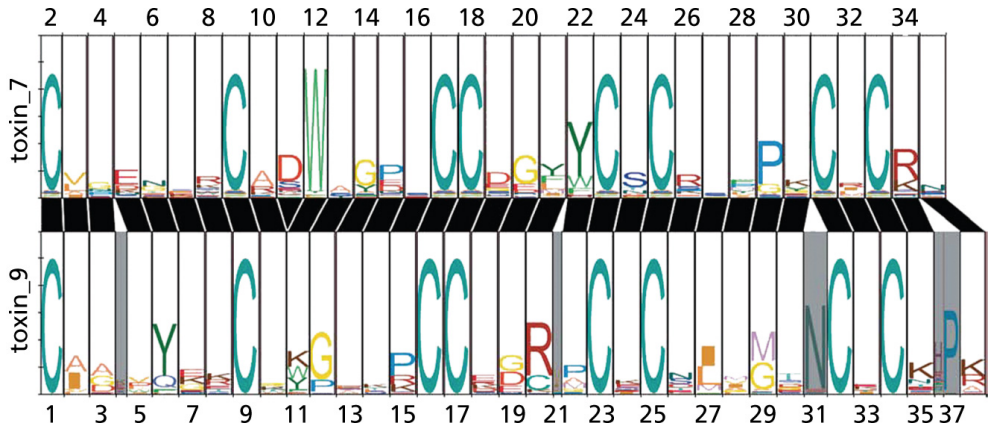
*Example of a HMM profile for the **Toxin_7 family**, from the **Pfam database***

Alignment of HMM profiles

The alignment of two HMM profiles is actually the alignment of two alignments; in it the gap scoring is position-dependent

In a possible approach, one multipe alignment is firstly reduced to a profile HMM, then a modification of the Viterbi algorithm is used to find the most probable set of paths which emit the other alignment (to get the overall probability for the alignment the probabilities for each sequence path must be multiplied)

HHsearch aligns two profile HMMs and is designed to identify very remote homologs; it also uses a variant of the Viterbi algorithm to find the alignment with the best score



Simplified visualization of the alignment of two HMMs (from Pfam) using logos to illustrate the emission probabilities at each match state

Tools > Multiple Sequence Alignment > Clustal Omega

Multiple Sequence Alignment

Clustal Omega is a new multiple sequence alignment program that uses seeded guide trees and HMM profile-profile techniques to generate alignments between **three or more** sequences. For the alignment of two sequences please instead use our pairwise sequence alignment tools.

Important note: This tool can align up to 4000 sequences or a maximum file size of 4 MB.

STEP 1 - Enter your input sequences

Enter or paste a set of

PROTEIN

sequences in any supported format:

Or, upload a file: Choose File no file selected

Use a example sequence | Clear sequence | See more example inputs

https://www.ebi.ac.uk/Tools/msa/clustalo/


gi|186910296|ref|NP_001119574.1| VPEKKTPKSPVGVQPILNEHTFCAGMSK------YQEDTCYGDAGSAFAVHDL------- 305
gi|4826762|ref|NP_005134.1| VPEKKTPKSPVGVQPILNEHTFCAGMSK------YQEDTCYGDAGSAFAVHDL------- 364
gi|21264363|ref|NP_006601.2| Y----------PRGSVTANMLCAGLES------GGKDSCRGDSGGALVFLDS------- 642
gi|21264357|ref|NP_001870.3| ----------KKVTRDMICAGEKE------GGKDACAGDSGGPMVTLNR------- 655
gi|21264359|ref|NP_624302.1| ----------GNYSVTENMFCAGYYE------GGKDTCLGDSGGAFVIFDD------- 673
gi|4502495|ref|NP_001725.1| TADA--------EAYVFTPNMICAGGEK-------GMDSCKGDSGGAFAVQDP------- 641
gi|66347875|ref|NP_001724.3| ---N--------RMDVFSQNMFCAGHPS------LKQDACQGDSGGVFAVRDP------- 663
gi|289547636|ref|NP_057630.2| ---Q--------RPEVFSDNMFCVGDET------QRHSVCQGDSGSVYVVWDN------- 445
gi|4758502|ref|NP_004123.1| ----------MIDDSMICAGNLQKP-----GQDTCQGDSGGPLTCE--------- 516
gi|295054188|ref|NP_001171131.1| ----------MIDDSMICAGNLQKP-----GQDTCQGDSGGPLTCE--------- 490
gi|4504383|ref|NP_001519.1| ----------DISPNMLCAGYFDC------KSDACQGDSGGPLACE--------- 605
gi|14702169|ref|NP_127509.1| ----------TVTDNMLCAGDTRSGGPQANLHDACQGDSGGPLVCL--------- 474
gi|4505861|ref|NP_000921.1| ----------TVTDNMLCAGDTRSGGPQANLHDACQGDSGGPLVCL--------- 520
gi|4505863|ref|NP_002649.1| ----------EVTTKMLCAADPQW------KTDSCQGDSGGPLVCS--------- 383
gi|222537759|ref|NP_001138503.1| ----------EVTTKMLCAADPQW------KTDSCQGDSGGPLVCS--------- 366
gi|119392081|ref|NP_000195.2| ----------FYEKEMECAGTYD------GSIDACKGDSGGPLVCMDA------- 534
gi|4503635|ref|NP_000497.1| ----------RITDNMFCAGYKPDEG---KRGDACEGDSGGPFVMKSP------- 577
gi|4506115|ref|NP_000303.1| ----------MVSENMLCAGILG------DRQDACEGDSGGPMVASF-------- 410
gi|10518503|ref|NP_062562.1| ----------NITEYMFCAGYSD------GSKDSCKGDSGGPHATHY-------- 390
gi|4503645|ref|NP_000122.1| ----------NITEYMFCAGYSD------GSKDSCKGDSGGPHATHY-------- 412
gi|4503625|ref|NP_000495.1| ----------IITQNMFCAGYDT------KQEDACQGDSGGPHVTRF-------- 427
gi|4503649|ref|NP_000124.1| ----------TIYNNMFCAGFHE------GGRDSCQGDSGGPHVTEV-------- 419
gi|32698940|ref|NP_872365.1| ----------IIPNTSFCAGDED------GAFDTCRGDSGGPLMCYLP------- 277
gi|148231605|ref|NP_937828.2| ----------ISGKTFLCTGFPD------GGRDACQGDSGGSLMCRNK------- 249
gi|300244530|ref|NP_003610.2| ----------RFTGRMLCAGNLHEH----KRVDSCQGDSGGPLMCERP------- 834
gi|205360943|ref|NP_005647.3| ----------LITPAMICAGFLQ------GNVDSCQGDSGGPLVTSK-------- 449
gi|227499990|ref|NP_001128571.1| ----------LITPAMICAGFLQ------GNVDSCQGDSGGPLVTSK-------- 486
gi|13173471|ref|NP_076927.1| ----------IISPSMLCAGYLT------GGVDSCQGDSGGPLVCQE-------- 409
gi|33667063|ref|NP_892018.1| ----------QISSRMLCAGFPQ------GGVDSCSGDAGGPLACREP------- 1018
gi|4505881|ref|NP_000292.1| ----------RVQSTELCAGHLA------GGTDSCQGDSGGPLVCFE-------- 768
gi|116292750|ref|NP_005568.2| ----------ICAEHLA------RGTDSCQGDSGGPLVCFE-------- 1998
gi|58331209|ref|NP_001962.3| ----------TVKNTMVCAGGD-------GVRSGCQGDSGGPLHCLV-------- 214
gi|62526043|ref|NP_009203.2| ----------RVKKTMVCAGGD-------GVISACNGDSGGPLNCQLE------- 225
gi|15559207|ref|NP_254275.1| ----------SVKTSMICAGGD-------GVISSCNGDSGGPLNCQAS------- 225
gi|58331211|ref|NP_056933.2| ----------TVKTNMICAGGD-------GVICTCNGDSGGPLNCQAS------- 225
gi|110815798|ref|NP_899234.2| ----------GITEKMICAGFAA-S----GEKDFCQGDSGGPLVCRHE------- 772
gi|4503137|ref|NP_001898.1| ----------SITDSMICAGGA--------GASSCQGDSGGPLVCQKG------- 223
gi|118498341|ref|NP_001897.4| ----------RITDVMICAGAS--------GVSSCMGDSGGPLVCQKD------- 222
gi|118498350|ref|NP_001020371.3| ----------RITDVMICAGAS--------GVSSCMGDSGGPLVCQKD------- 222

Clustal Omega

Input form | Web services | Help & Documentation | Bioinformatics Tools FAQ

Tools > Multiple Sequence Alignment > Clustal Omega

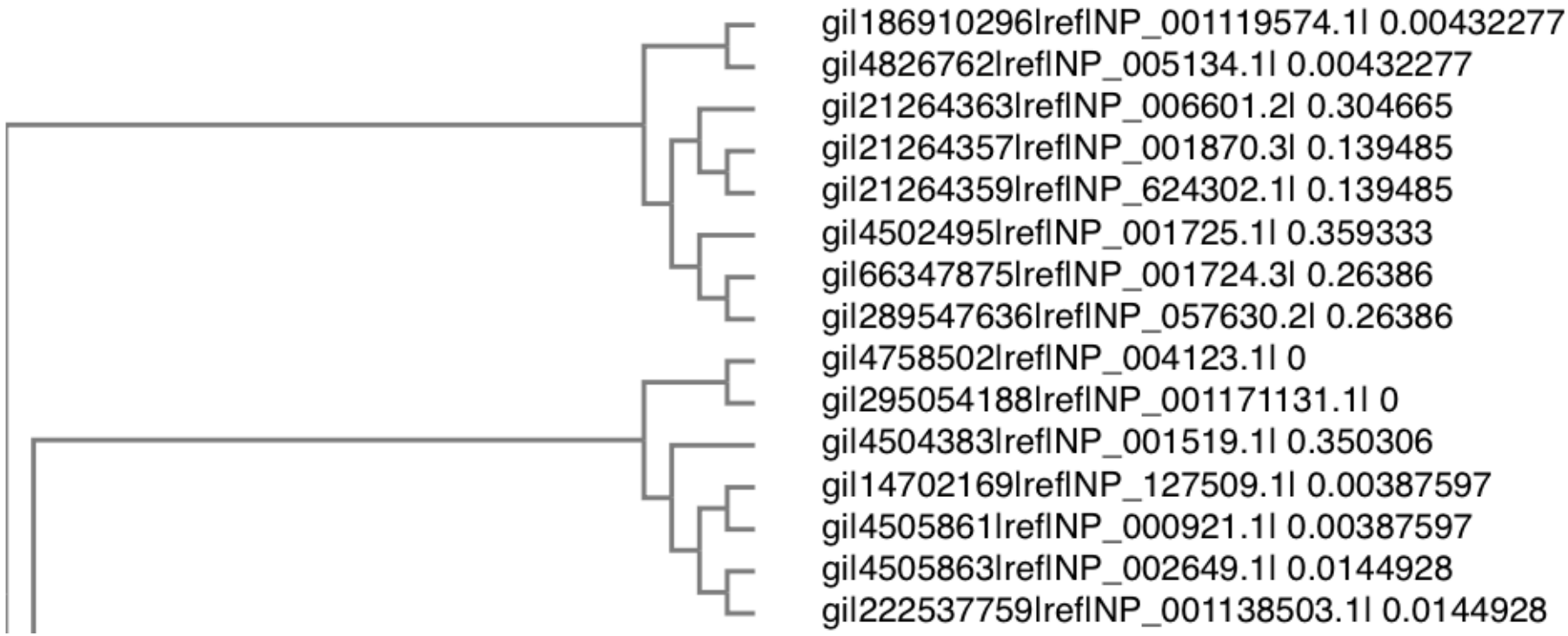
Results for job clustalo-I20220829-154434-0539-1190104-p2m

Alignments | Result Summary | Guide Tree | Phylogenetic Tree | Results Viewers | Submission Details

Download Guide Tree Data

Phylogram

Branch length: Cladogram Real



MSAs: other approaches

DIALIGN is a local alignment method

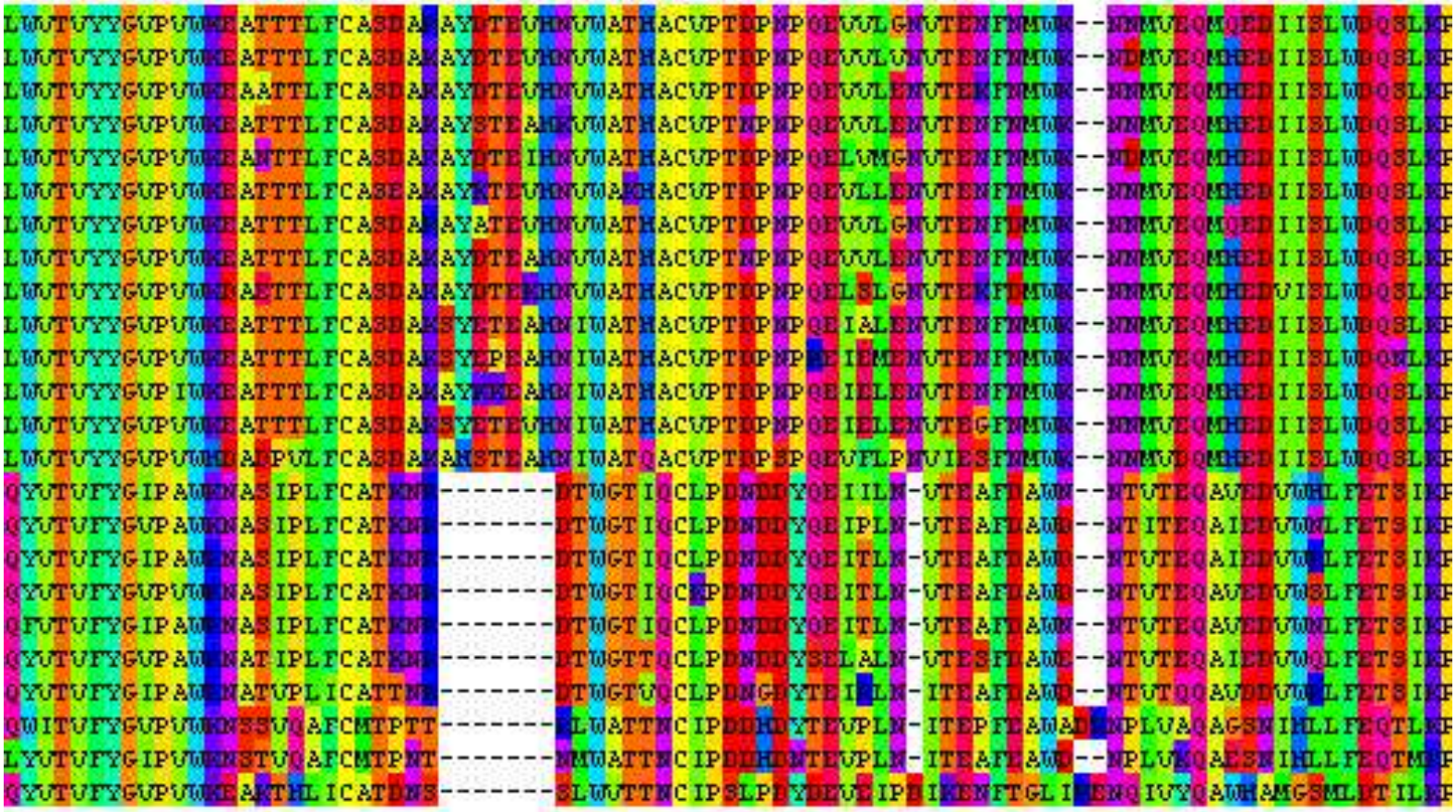
It constructs pairwise ad multiple alignments by comparing whole ungapped segments several residues long

The alignment is then constructed from pairs of equal-length gap-free segments (diagonals)

Many diagonals will overlap and the program has to find a set of diagonals which can be combined into one consistent alignment

DIALIGN is suitable for sequences of moderate length

Multiple alignments



Two sequences whisper, many homologous sequences talk loud

A. Lesk

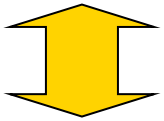
Why are multiple alignments so important?

Because

they allow to obtain accurate alignments

outline positions subjected to evolutionary pressure

provide relevant functional/structural insight



Provide information on the evolutionary process

Why are multiple alignments so important?

```
(A) p110α          TFILGIGDRHNSNIMVKDDG-QLFHIDFGHFLDHKKKKFGYKRERVPFVLT--QDFLIVI 142
    cAMP-kinase    QIVLTFEYLHSLDLIYRDLKPENLLIDQQGYIQVTDFGFAKRVKGRTWXLCGTPEYLAPE 179
```

pairwise alignment of the catalytic domains of PI3-kinase p110α and a cAMP-dependent protein kinase

```
(B) p110β          SYVLGIG----------DRHSDNINVKKTGQLFHIDFGHILGNFKSKFGIKRERVPFILT 136
    p110δ          TYVLGIG----------DRHSDNIMIRESGQLFHIDFGHFLGNFKTKFGINRERVPFILT 136
    p110α          TFILGIG----------DRHNSNIMVKDDGQLFHIDFGHFLDHKKKKFGYKRERVPFVLT 135
    p110γ          TFVLGIG----------DRHNDNIMITETGNLFHIDFGHILGNYKSFLGINKERVPFVLT 135
    p110_dicti     TYVLGIG----------DRHNDNLMVTKGGRLFHIDFGHFLGNYKKKFGFKRERAPFVFT 135
    cAMP-kinase    QIVLTFEYLHSLDLIYRDLKPENLLIDQQGYIQVTDFGFAKRVKGRTWXLCG--TPEYLA 177
```

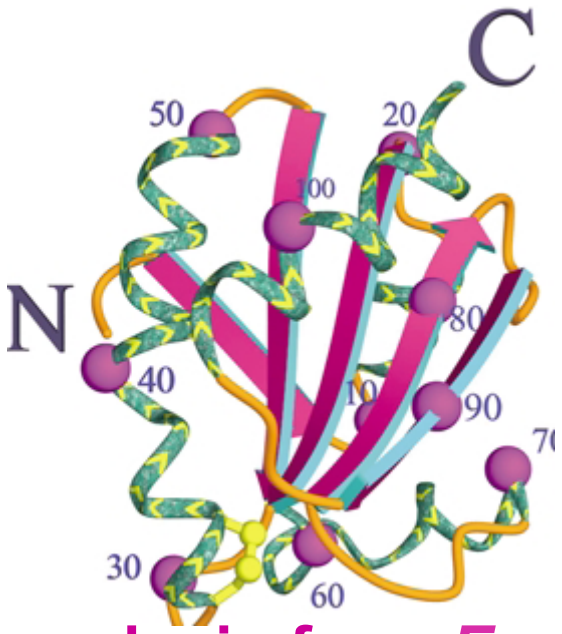
their ClustalW alignment with other PI3-kinases

In the multiple alignment, the functionally important residues (highlighted in green) are correctly aligned

One more example: thioredoxins

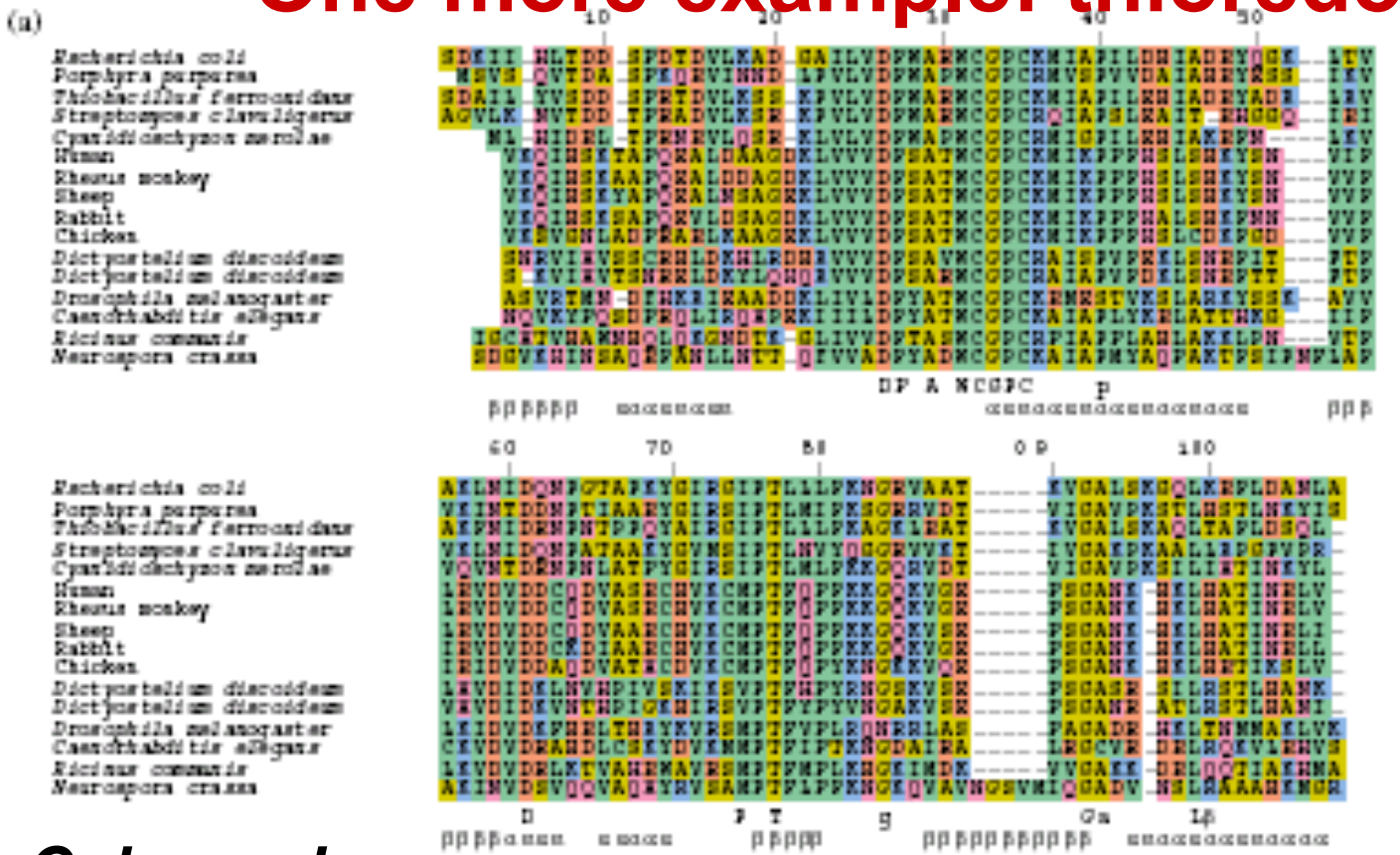
Involved in cell proliferation, blood coagulation, insulin degradation, enzymatic regulation etc.

Fold α/β: β sheet of five strands flanked by α-helices



Thioredoxin from *E. coli*

One more example: thioredoxins



Color code:

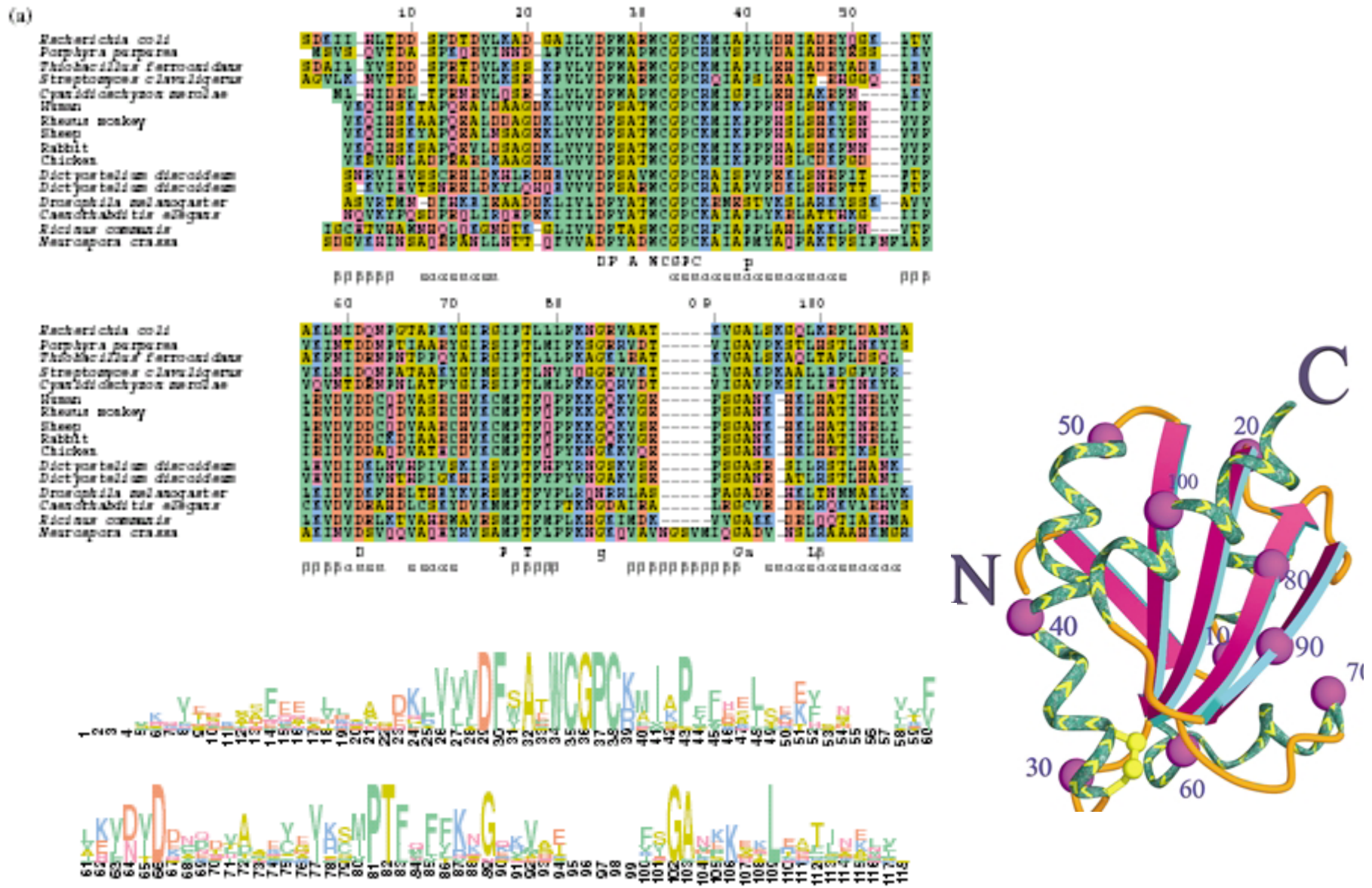
Gly, Ala, Ser, Thr : small

Cys, Val, Ile, Leu, Pro, Phe, Tyr, Met, Trp : hydrophobic

Asn, Gln, His : polar

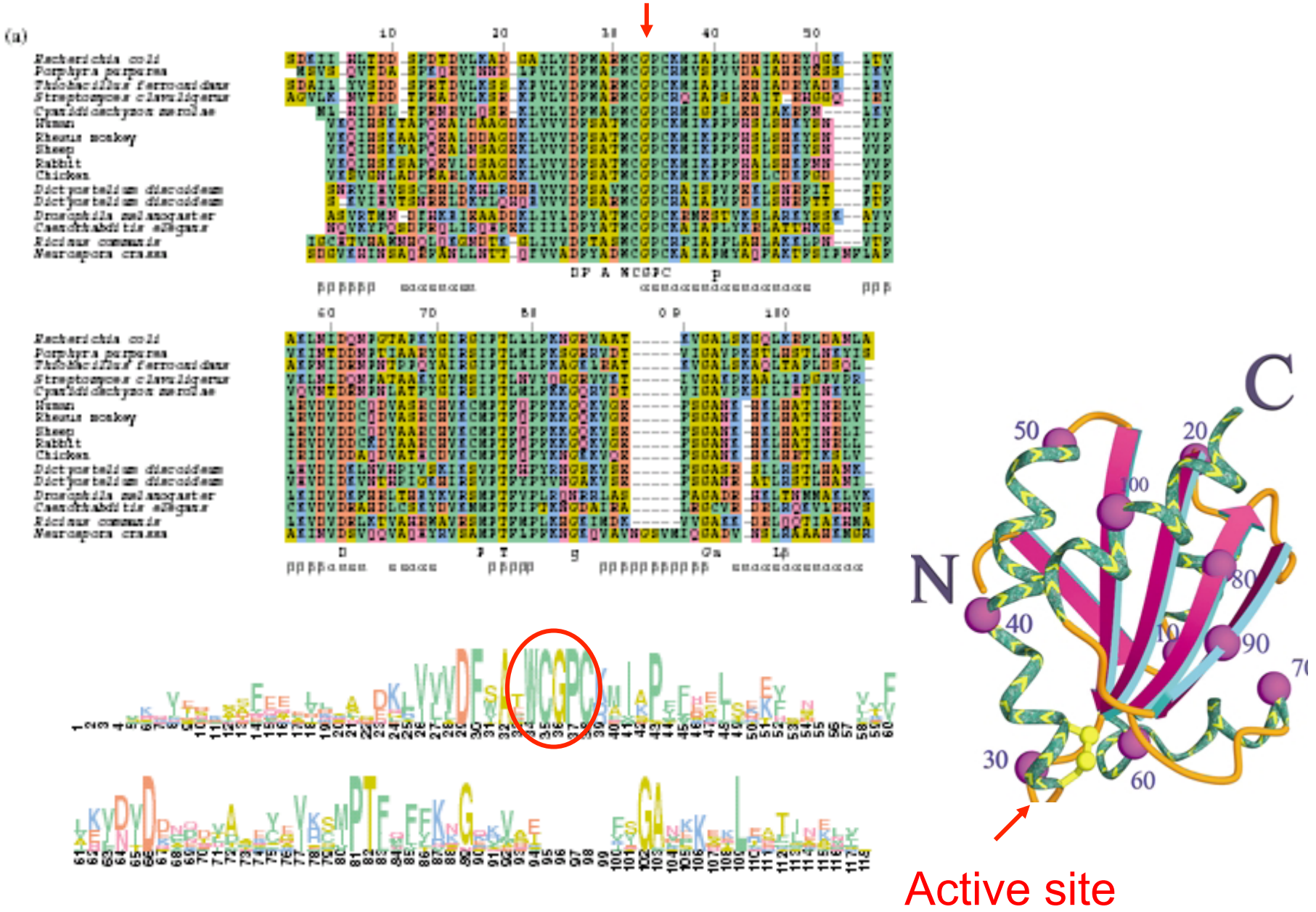
Asp, Glu : negatively charged

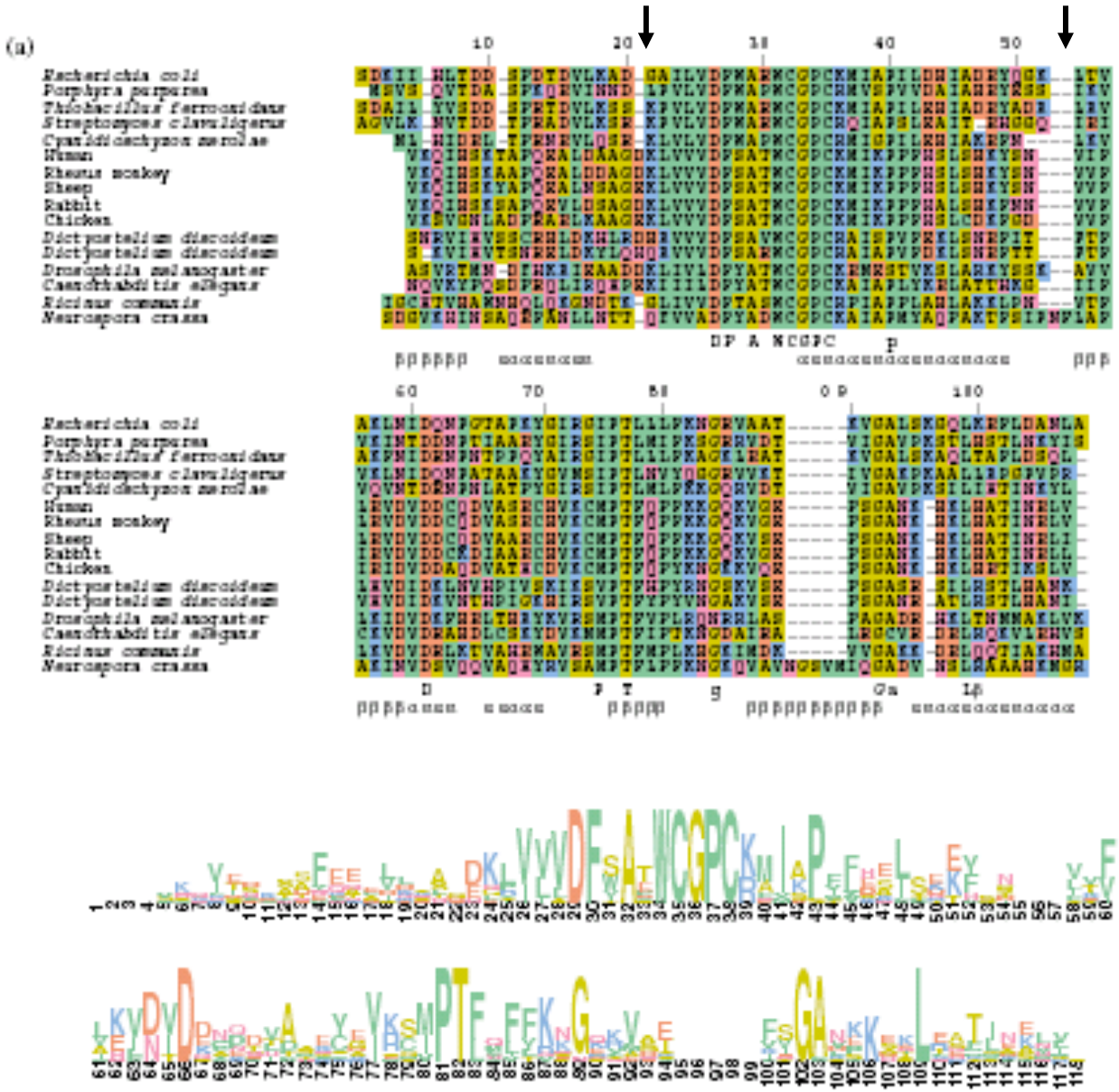
Lys, Arg : positively charged



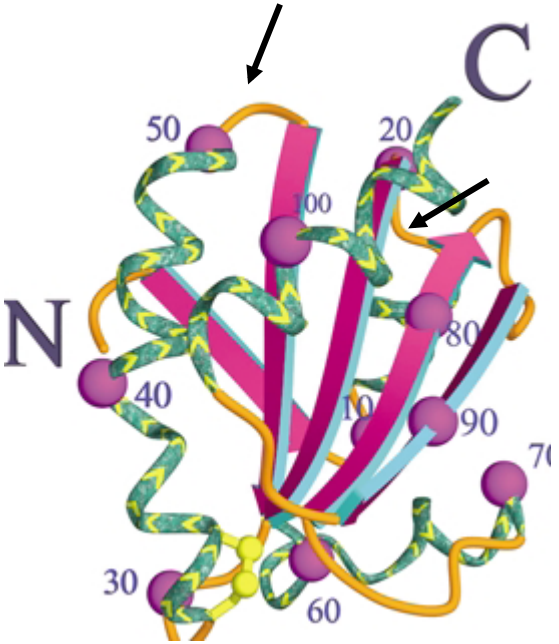
Profile

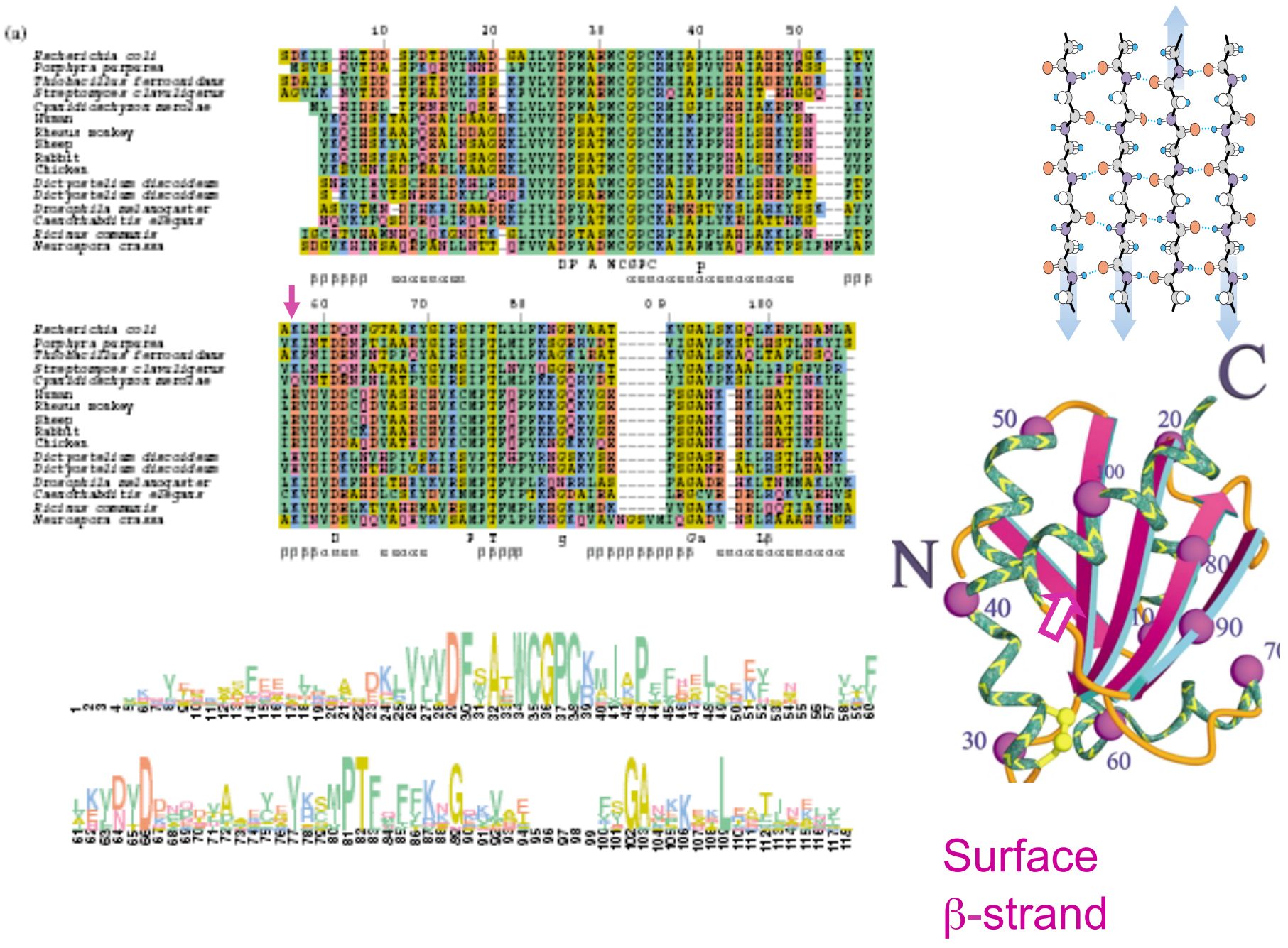
sequence logo representation

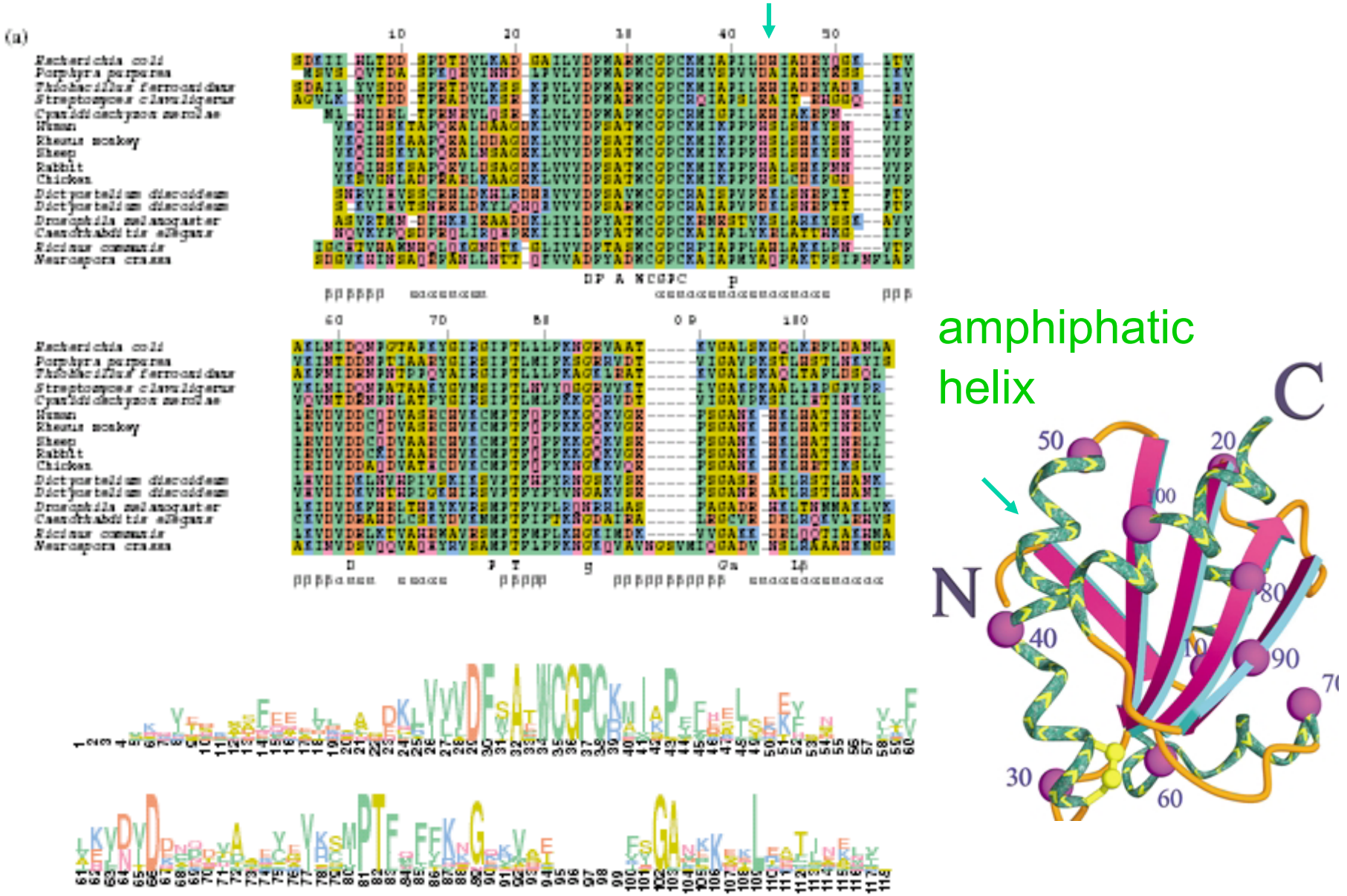

(a)
Escherichia coli
Porphyra purpurea
Thiobacillus ferrooxidans
Streptomyces clavuligerus
Cyanidioschyzon merolae
Human
Rhesus monkey
Sheep
Rabbit
Chicken
Dictyostelium discoideum
Dictyostelium discoideum
Drosophila melanogaster
Caenorhabditis elegans
Ricinus communis
Neurospora crassa
N
C
Active site

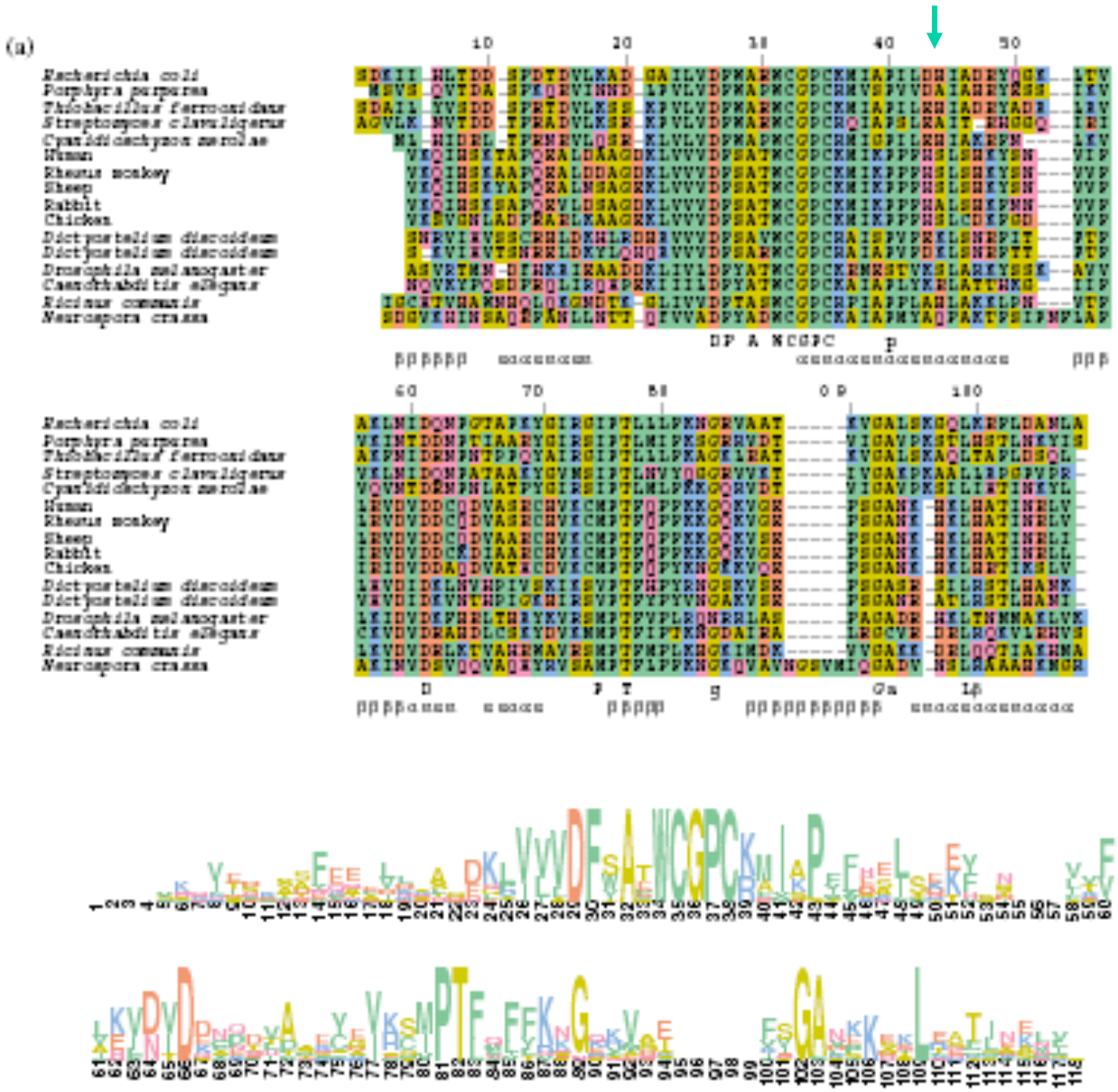


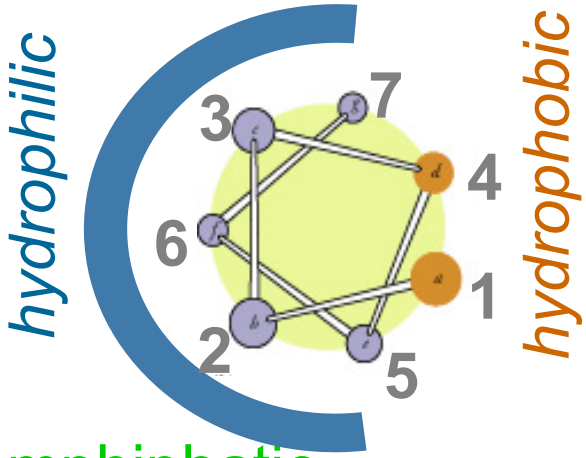
Surface (exposed) loops

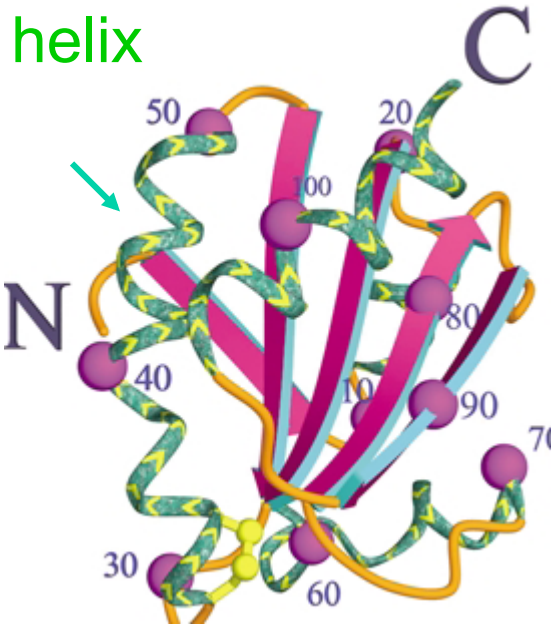



(a)
C
N
Surface
β-strand


(a)
amphiphatic
helix
C
N
50
20
100
80
40
90
30
60




hydrophilic
hydrophobic
amphiphatic
helix
1
2
3
4
5
6
7


C
N
10
20
30
40
50
60
70
80
90
100

Lessons 5 & 6. Content

1. Alignment algorithms. There are exact and heuristic ones. We choose one or the other depending on the wanted application

2. Multiple alignments. They contain precious information about the evolutionary path. Non-exact methods are used to obtain them. HMMs can be applied. They are extremely informative on the structure and function of corresponding proteins