Using Genetic Algorithms for Pairwise and Multiple Sequence Alignments

Overview

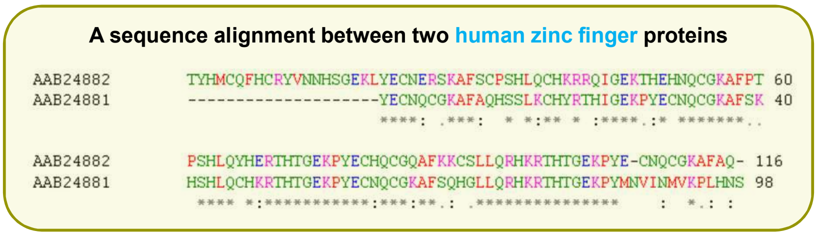
- Introduction
- Methods
- Results
- Discussion

SECTION 0

Sequence Alignment

What Is Sequence Alignment?

- Sequence alignment in bioinformatics
  - **Compare** the sequences of DNA, RNA and protein



A sequence alignment between two human zinc finger proteins

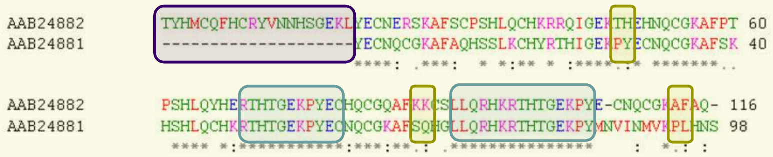
```
AAB24882      TYHMCQFHCRYVNNHSGEKLYECNERSKAFSCPSHLQCHKRRQIGEKTHEHNQCGKAFPT 60
AAB24881      --------------------YECNQCGKAFAQHSSLKCHYRTHIGEKPYECNQCGKAFSK 40
                                  ****: .***:  * *:** * :****.:* *******..

AAB24882      PSHLQYHERTHTGEKPYECHQCGQAFKKCSLLQRHKRTHTGEKPYE-CNQCGKAFAQ- 116
AAB24881      HSHLQCHKRTHTGEKPYECNQCGKAFSQHGLLQRHKRTHTGEKPYMNVINMVKPLHNS 98
               **** *:***********:***:**.: .**************    :  *.: :
```

- Rows → **sequences**: DNA/RNA || protein   **chromosome**
- Columns → **residues**: nucleotide || amino acid   **gene**

Why Sequence Alignment?

- Sequence alignment in bioinformatics
  - Identify regions of **similarity**
    → functional, structural, or evolutionary **relationships**

A sequence alignment between two human zinc finger proteins



```
AAB24882      TYHMCQFHCRYVNNHSGEKLYECNERSKAFSCPSHLQCHKRRQIGEKTHEHNQCGKAFPT 60
AAB24881      --------------------YECNQCGKAFAQHSSLKCHYRTHIGEKPYECNQCGKAFSK 40
                                  ****:  .***:   *  *:**  *  :****.:*  *******..

AAB24882      PSHLQYHERTHTGEKPYECHQCGQAFKKCSLLQRHKRTHTGEKPYE-CNQCGKAFAQ- 116
AAB24881      HSHLQCHKRTHTGEKPYECNQCGKAFSQHGLLQRHKRTHTGEKPYMNVINMVKPLHNS 98
               ****  *:***********:***:**.:  .***************      :   *.:  :
```

- Identity: **conserved** region → structural or functional **importance**
- Substitution: point **mutation**
- Gap: insertion/deletion **mutation**

} **diverge** from a common ancestor

How To Solve?

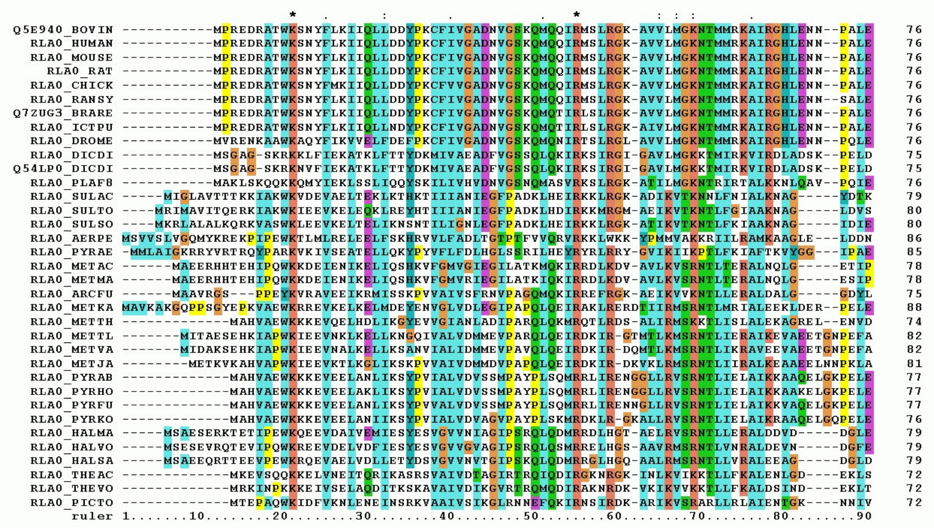
- Very **short** or very similar sequences
  → can be aligned by hand
- **Lengthy**, highly variable or extremely **numerous** sequences
  → cannot be aligned solely by human effort

human knowledge
construct algorithms to produce high-quality sequence alignments

- Dynamic programming
  - Slow but **formally optimizing**
- Heuristic or probabilistic methods
  - **Efficient** for large scale search

Multiple Sequence Alignment

First 90 positions of a protein multiple sequence alignment of instances of the acidic ribosomal protein P0 (L10E) from several organisms, from Wikipedia.

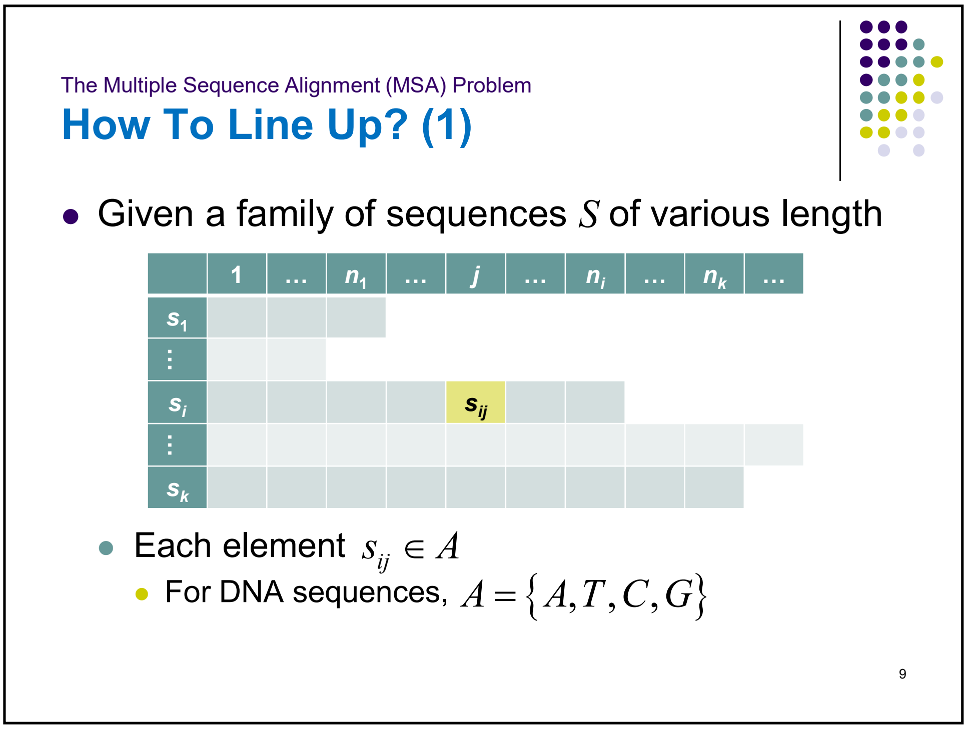


```
                                 *   .      :       .          .   *         : : :      .
Q5E940_BOVIN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_HUMAN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_MOUSE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
    RLA0_RAT -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_CHICK -----------MPREDRATWKSNYFMKIIQLLDDYPKCFVVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_RANSY -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--SALE    76
Q7ZUG3_BRARE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQTIRLSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_ICTPU -----------MPREDRATWKSNYFLKIIQLLNDYPKCFIVGADNVGSKQMQTIRLSLRGK-AIVLMGKNTMMRKAIRGHLENN--PALE    76
  RLA0_DROME -----------MVRENKAAWKAQYFIKVVELFDEFPKCFIVGADNVGSKQMQNIRTSLRGL-AVVLMGKNTMMRKAIRGHLENN--PQLE    76
  RLA0_DICDI -----------MSGAG-SKRKKLFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIRDLADSK--PELD    75
Q54LP0_DICDI -----------MSGAG-SKRKNVFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIRDLADSK--PELD    75
  RLA0_PLAF8 -----------MAKLSKQQKKQMYIEKLSSLIQQYSKILIVHVDNVGSNQMASVRKSLRGK-ATILMGKNTRIRTALKKNLQAV--PQIE    76
  RLA0_SULAC -----MIGLAVTTTKKIAKWKVDEVAELTEKLKTHKTIIIANIEGFPADKLHEIRKKLRGK-ADIKVTKNNLFNIALKNAG-----YDTK    79
  RLA0_SULTO ----MRIMAVITQERKIAKWKIEEVKELEQKLREYHTIIIANIEGFPADKLHDIRKKMRGM-AEIKVTKNTLFGIAAKNAG-----LDVS    80
  RLA0_SULSO ----MKRLALALKQRKVASWKLEEVKELTELIKNSNTILIGNLEGFPADKLHEIRKKLRGK-ATIKVTKNTLFKIAAKNAG-----IDIE    80
  RLA0_AERPE MSVVSLVGQMYKREKPIPEWKTLMLRELEELFSKHRVVLFADLTGTPTFVVQRVRKKLWKK-YPMMVAKKRIILRAMKAAGLE---LDDN    86
  RLA0_PYRAE -MMLAIGKRRYVRTRQYPARKVKIVSEATELLQKYPYVFLFDLHGLSSRILHEYRYRLRRY-GVIKIIKPTLFKIAFTKVYGG---IPAE    85
  RLA0_METAC ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVGIEGILATKMQKIRRDLKDV-AVLKVSRNTLTERALNQLG-----ETIP    78
  RLA0_METMA ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVRIEGILATKIQKIRRDLKDV-AVLKVSRNTLTERALNQLG-----ESIP    78
  RLA0_ARCFU ------MAAVRGS---PPEYKVRAVEEIKRMISSKPVVAIVSFRNVPAGQMQKIRREFRGK-AEIKVVKNTLLERALDALG-----GDYL    75
  RLA0_METKA MAVKAKGQPPSGYEPKVAEWKRREVKELKELMDEYENVGLVDLEGIPAPQLQEIRAKLRERDTIIRMSRNTLMRIALEEKLDER--PELE    88
  RLA0_METTH -------------MAHVAEWKKKEVQELHDLIKGYEVVGIANLADIPARQLQKMRQTLRDS-ALIRMSKKTLISLALEKAGREL--ENVD    74
  RLA0_METTL -------MITAESEHKIAPWKIEEVNKLKELLKNGQIVALVDMMEVPARQLQEIRDKIR-GTMTLKMSRNTLIERAIKEVAEETGNPEFA    82
  RLA0_METVA -------MIDAKSEHKIAPWKIEEVNALKELLKSANVIALIDMMEVPAVQLQEIRDKIR-DQMTLKMSRNTLIKRAVEEVAEETGNPEFA    82
  RLA0_METJA --------METKVKAHVAPWKIEEVKTLKGLIKSKPVVAIVDMMDVPAPQLQEIRDKIR-DKVKLRMSRNTLIIRALKEAAEELNNPKLA    81
  RLA0_PYRAB -------------MAHVAEWKKKEVEELANLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAQELGKPELE    77
  RLA0_PYRHO -------------MAHVAEWKKKEVEELAKLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAKELGKPELE    77
  RLA0_PYRFU -------------MAHVAEWKKKEVEELANLIKSYPVVALVDVSSMPAYPLSQMRRLIRENNGLLRVSRNTLIELAIKKVAQELGKPELE    77
  RLA0_PYRKO -------------MAHVAEWKKKEVEELANIIKSYPVIALVDVAGVPAYPLSKMRDKLR-GKALLRVSRNTLIELAIKRAAQELGQPELE    76
  RLA0_HALMA -----MSAESERKTETIPEWKQEEVDAIVEMIESYESVGVVNIAGIPSRQLQDMRRDLHGT-AELRVSRNTLLERALDDVD-----DGLE    79
  RLA0_HALVO -----MSESEVRQTEVIPQWKREEVDELVDFIESYESVGVVGVAGIPSRQLQSMRRELHGS-AAVRMSRNTLVNRALDEVN-----DGFE    79
  RLA0_HALSA -----MSAEEQRTTEEVPEWKRQEVAELVDLLETYDSVGVVNVTGIPSKQLQDMRRGLHGQ-AALRMSRNTLLVRALEEAG-----DGLD    79
  RLA0_THEAC -------------MKEVSQQKKELVNEITQRIKASRSVAIVDTAGIRTRQIQDIRGKNRGK-INLKVIKKTLLFKALENLGD----EKLS    72
  RLA0_THEVO -------------MRKINPKKKEIVSELAQDITKSKAVAIVDIKGVRTRQMQDIRAKNRDK-VKIKVVKKTLLFKALDSIND----EKLT    72
  RLA0_PICTO -------------MTEPAQWKIDFVKNLENEINSRKVAAIVSIKGLRNNEFQKIRNSIRDK-ARIKVSRARLLRLAIENTGK----NNIV    72
       ruler 1.......10........20........30........40........50........60........70........80........90
```

The Multiple Sequence Alignment (MSA) Problem

Introduction

- Multiple sequence alignment
  - **Simultaneous** alignment of **many** nucleotide or amino acid sequences
  - **Line up** the characters in a set of strings in the **best possible** way
    - How to line up?
      - Insert gaps into the strings to make equal length
    - What is best?
      - Score an alignment of multiple sequences



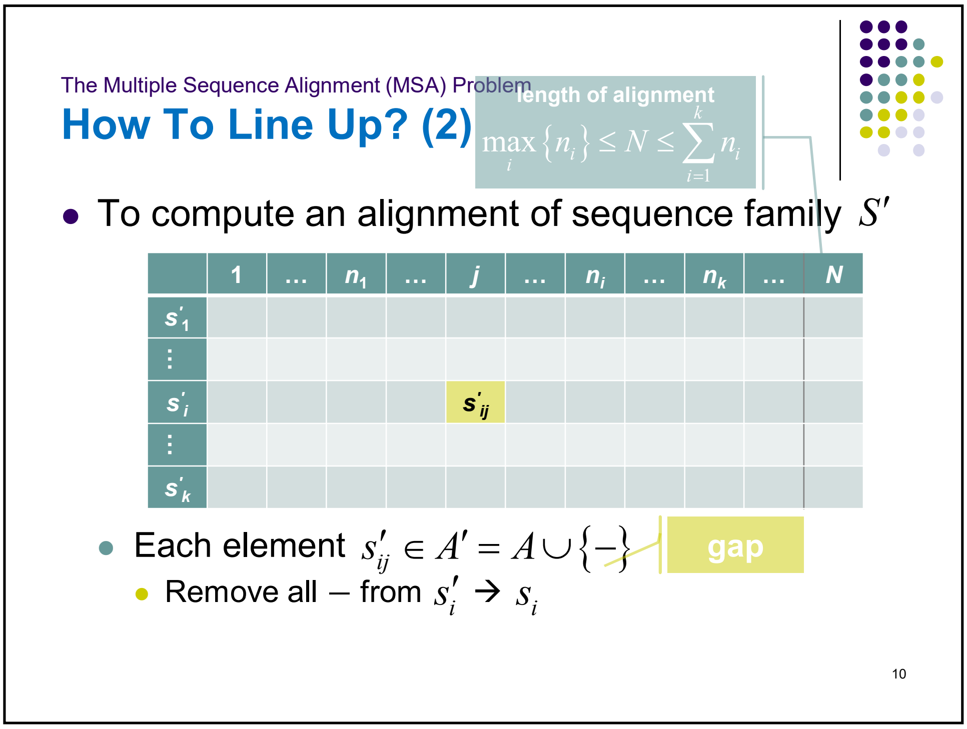
The Multiple Sequence Alignment (MSA) Problem

How To Line Up? (1)

- Given a family of sequences S of various length

	1	…	n_1	…	j	…	n_i	…	n_k	…
s_1										
⋮										
s_i					s_{ij}					
⋮										
s_k										

  - Each element $s_{ij} \in A$
    - For DNA sequences, $A = \{A, T, C, G\}$



The Multiple Sequence Alignment (MSA) Problem

How To Line Up? (2)

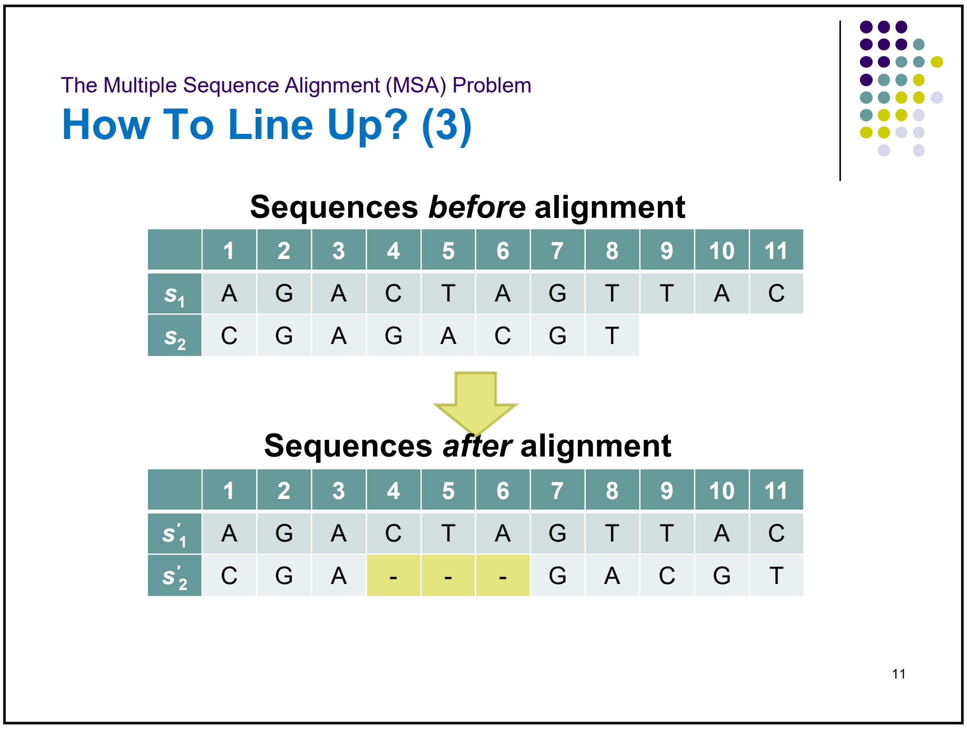
length of alignment

$\max_i \{n_i\} \le N \le \sum_{i=1}^{k} n_i$

- To compute an alignment of sequence family S'

	1	…	n_1	…	j	…	n_i	…	n_k	…	N
s'_1											
⋮											
s'_i					s'_{ij}						
⋮											
s'_k											

  - Each element $s'_{ij} \in A' = A \cup \{-\}$ gap
    - Remove all $-$ from s'_i → s_i



The Multiple Sequence Alignment (MSA) Problem

How To Line Up? (3)

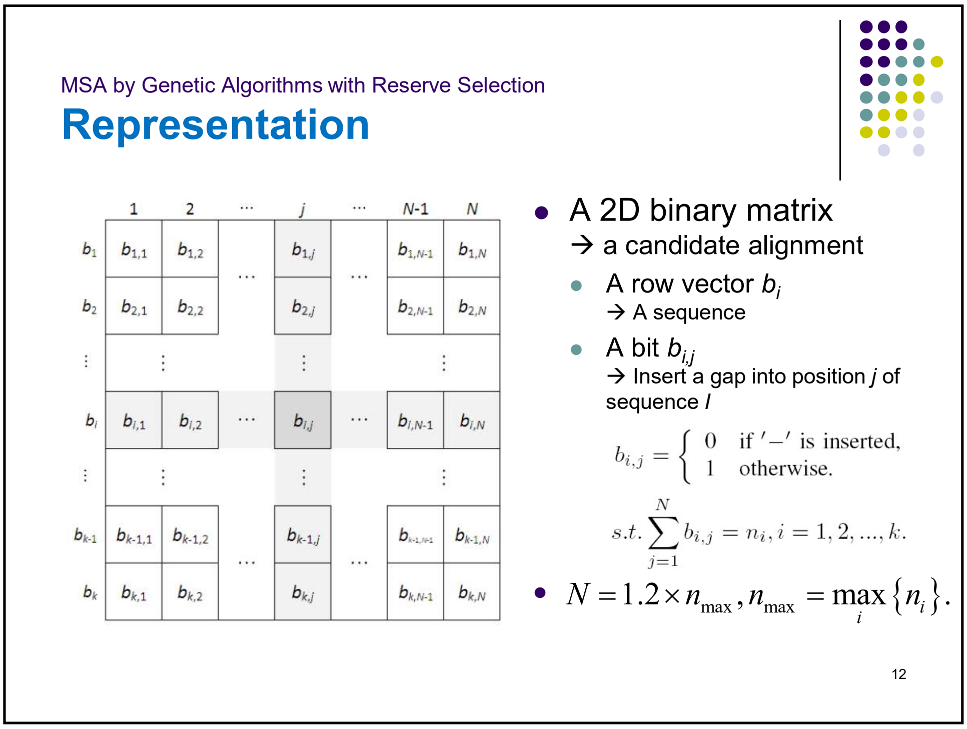
Sequences *before* alignment

	1	2	3	4	5	6	7	8	9	10	11
s_1	A	G	A	C	T	A	G	T	T	A	C
s_2	C	G	A	G	A	C	G	T			

Sequences *after* alignment

	1	2	3	4	5	6	7	8	9	10	11
s'_1	A	G	A	C	T	A	G	T	T	A	C
s'_2	C	G	A	-	-	-	G	A	C	G	T

11



MSA by Genetic Algorithms with Reserve Selection

Representation

$$\begin{array}{c|cccccc|}
 & 1 & 2 & \cdots & j & \cdots & N\text{-}1 & N \\
\hline
b_1 & b_{1,1} & b_{1,2} & \cdots & b_{1,j} & \cdots & b_{1,N-1} & b_{1,N} \\
b_2 & b_{2,1} & b_{2,2} & & b_{2,j} & & b_{2,N-1} & b_{2,N} \\
\vdots & & \vdots & & \vdots & & \vdots & \\
b_i & b_{i,1} & b_{i,2} & \cdots & b_{i,j} & \cdots & b_{i,N-1} & b_{i,N} \\
\vdots & & \vdots & & \vdots & & \vdots & \\
b_{k-1} & b_{k-1,1} & b_{k-1,2} & & b_{k-1,j} & & b_{k-1,N-1} & b_{k-1,N} \\
b_k & b_{k,1} & b_{k,2} & \cdots & b_{k,j} & \cdots & b_{k,N-1} & b_{k,N}
\end{array}$$

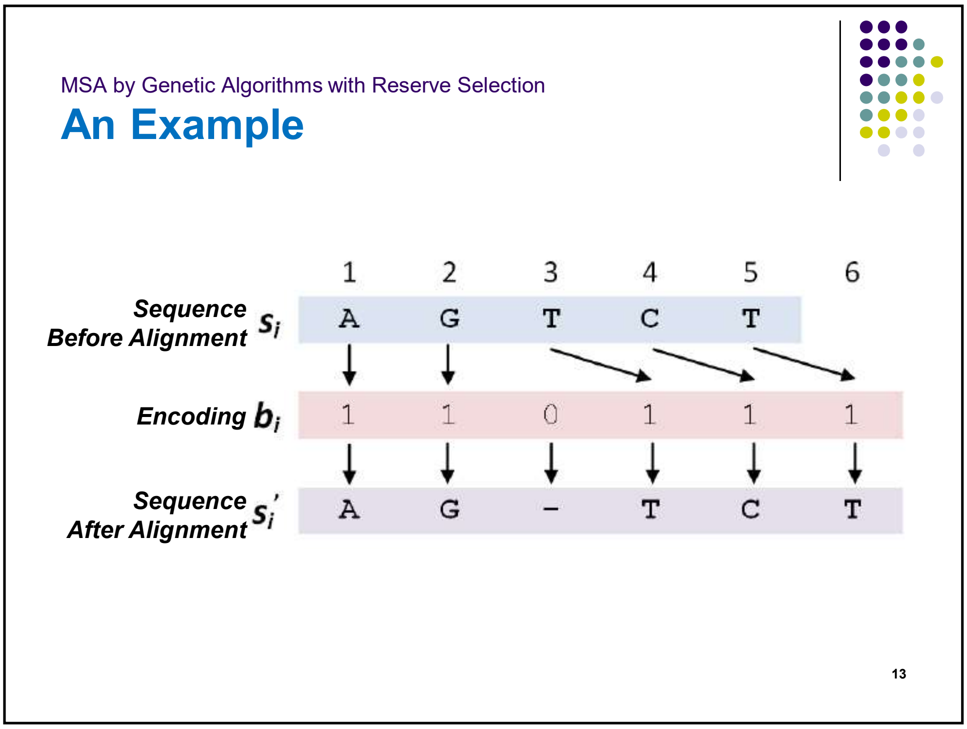
- A 2D binary matrix
  → a candidate alignment
  - A row vector b_i
    → A sequence
  - A bit $b_{i,j}$
    → Insert a gap into position *j* of sequence *I*

$$b_{i,j} = \begin{cases} 0 & \text{if } '-' \text{ is inserted,} \\ 1 & \text{otherwise.} \end{cases}$$

$$s.t. \sum_{j=1}^{N} b_{i,j} = n_i, i = 1, 2, ..., k.$$

- $N = 1.2 \times n_{\max}, n_{\max} = \max_i \{n_i\}.$

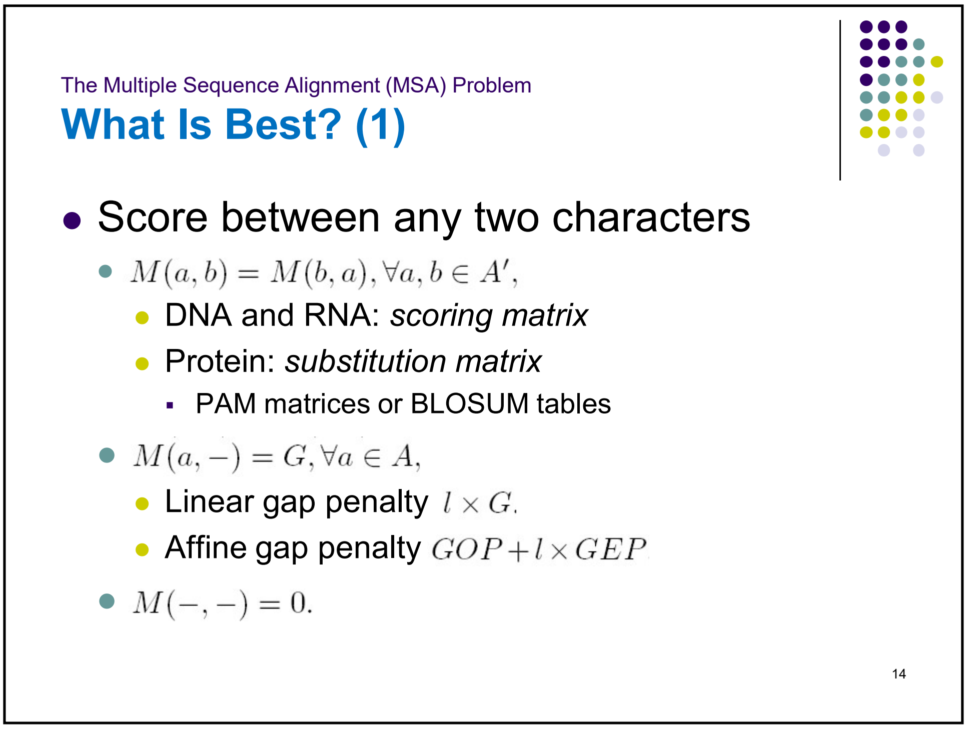
12



MSA by Genetic Algorithms with Reserve Selection

An Example

	1	2	3	4	5	6
Sequence Before Alignment s_i	A	G	T	C	T	
Encoding b_i	1	1	0	1	1	1
Sequence After Alignment s_i'	A	G	-	T	C	T



The Multiple Sequence Alignment (MSA) Problem

What Is Best? (1)

- Score between any two characters
  - $M(a,b) = M(b,a), \forall a,b \in A'$,
    - DNA and RNA: *scoring matrix*
    - Protein: *substitution matrix*
      - PAM matrices or BLOSUM tables
  - $M(a,-) = G, \forall a \in A$,
    - Linear gap penalty $l \times G$.
    - Affine gap penalty $GOP + l \times GEP$
  - $M(-,-) = 0$.

The Multiple Sequence Alignment (MSA) Problem

What Is Best? (2)

- Score between any two sequences

$$M(s'_i, s'_j) = \sum_{p=1}^{N} M(s'_{ip}, s'_{jp}).$$

- Score an alignment of multiple sequences
  - Sum-of-pairs score (*SP-score*)

$$M(S') = \sum_{1 \le i < j \le k} M(s'_i, s'_j).$$

- The MSA problem
  - To find an alignment that maximizes *SP-score*

The Multiple Sequence Alignment (MSA) Problem

What Is Best? (3)

- The alignment

	1	2	3	4	5	6	7	8	9	10	11
s'_1	A	G	A	C	T	A	G	T	T	A	C
s'_2	C	G	A	-	-	-	G	A	C	G	T

- Similarity matrix

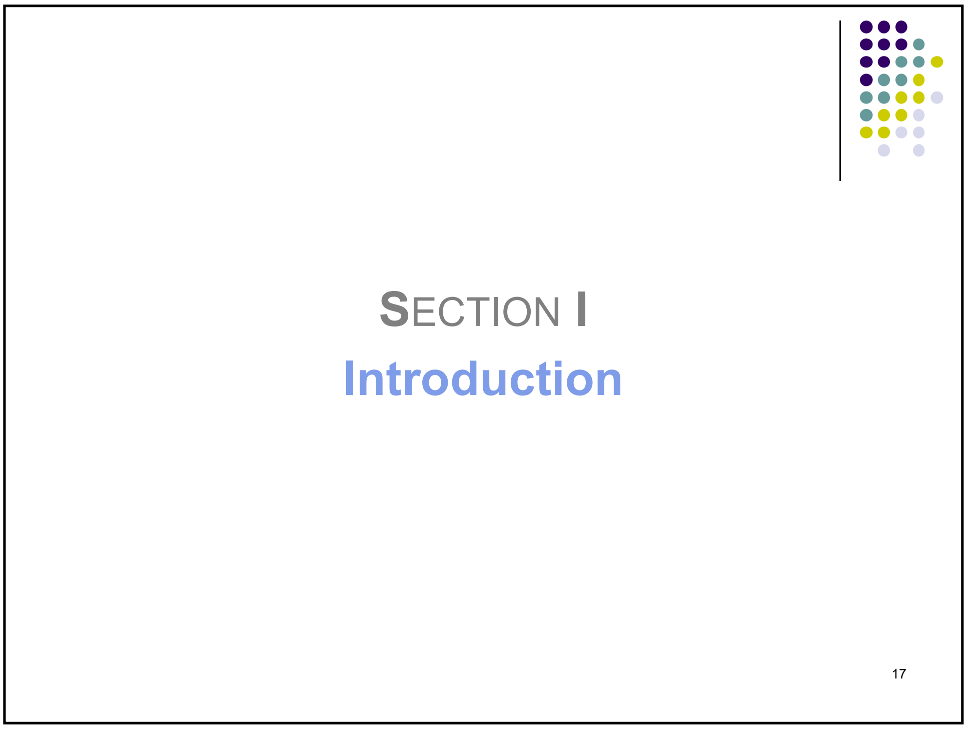
-	A	G	C	T
A	10	-1	-3	-4
G	-1	7	-5	-3
C	-3	-5	9	0
T	-4	-3	0	8

- Gap penalty

$$d = -5$$

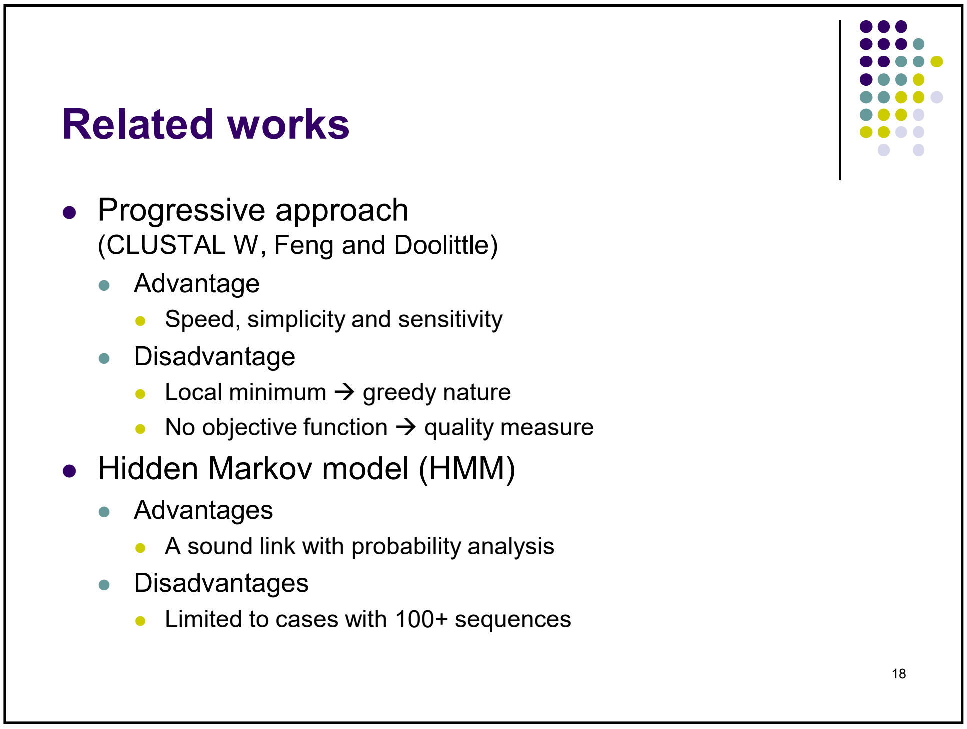
- The score

$$\begin{aligned} & M(s_1', s_2') \\ & = M(A,C) + M(G,G) + M(A,A) + 3 \times d + \\ & \ M(G,G) + M(T,A) + M(T,C) + M(A,G) + M(C,T) \\ & = (-3) + 7 + 10 + 3 \times (-5) + 7 + (-4) + 0 + (-1) + 0 \\ & = 1 \end{aligned}$$



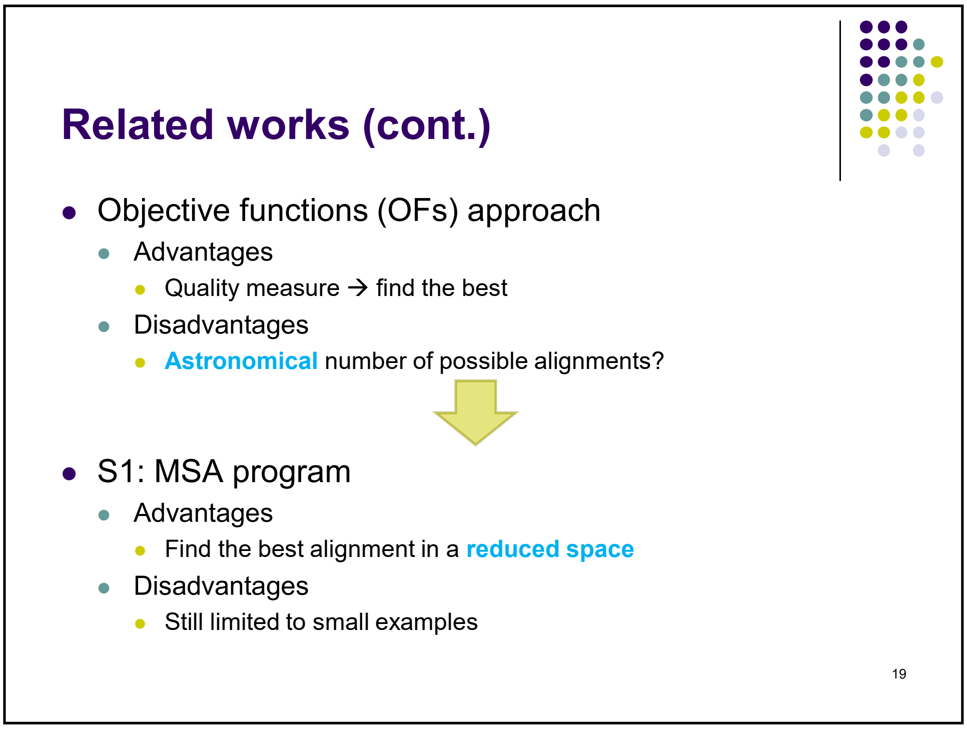
SECTION I

Introduction



Related works

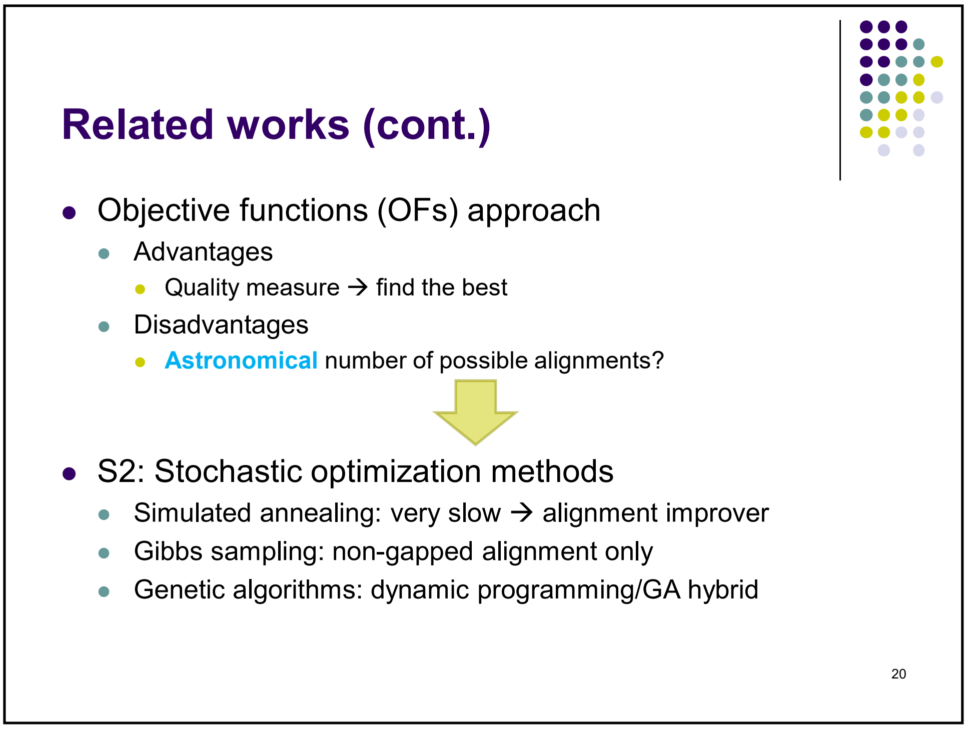
- Progressive approach
  (CLUSTAL W, Feng and Doolittle)
  - Advantage
    - Speed, simplicity and sensitivity
  - Disadvantage
    - Local minimum → greedy nature
    - No objective function → quality measure
- Hidden Markov model (HMM)
  - Advantages
    - A sound link with probability analysis
  - Disadvantages
    - Limited to cases with 100+ sequences



Related works (cont.)

- Objective functions (OFs) approach
  - Advantages
    - Quality measure → find the best
  - Disadvantages
    - **Astronomical** number of possible alignments?
- S1: MSA program
  - Advantages
    - Find the best alignment in a **reduced space**
  - Disadvantages
    - Still limited to small examples

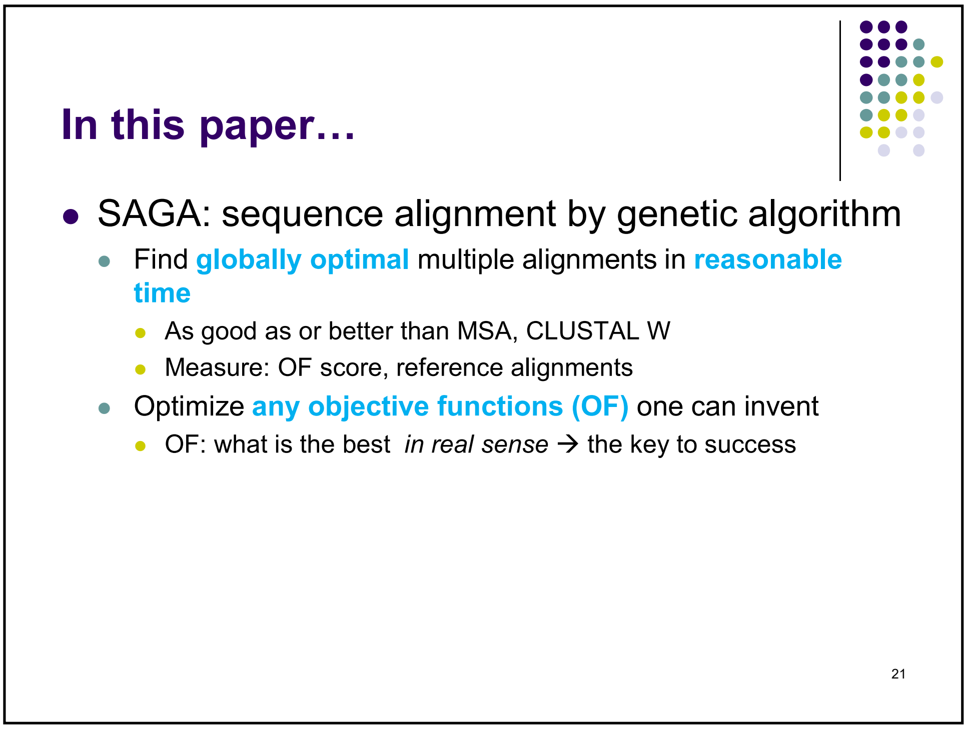
19



Related works (cont.)

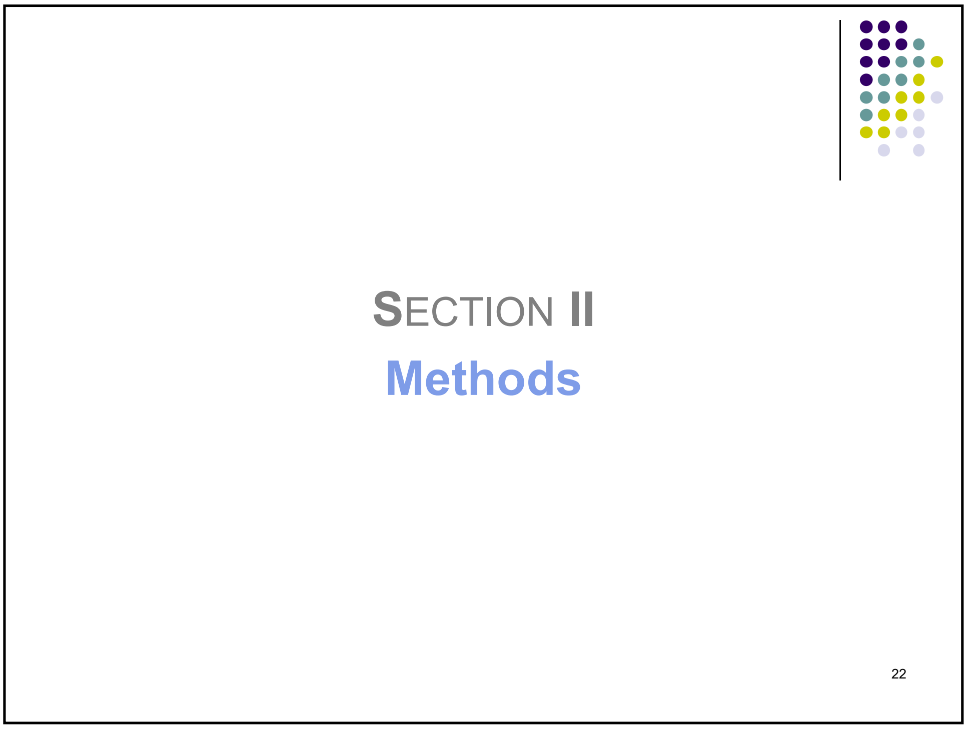
- Objective functions (OFs) approach
  - Advantages
    - Quality measure → find the best
  - Disadvantages
    - **Astronomical** number of possible alignments?
- S2: Stochastic optimization methods
  - Simulated annealing: very slow → alignment improver
  - Gibbs sampling: non-gapped alignment only
  - Genetic algorithms: dynamic programming/GA hybrid

20



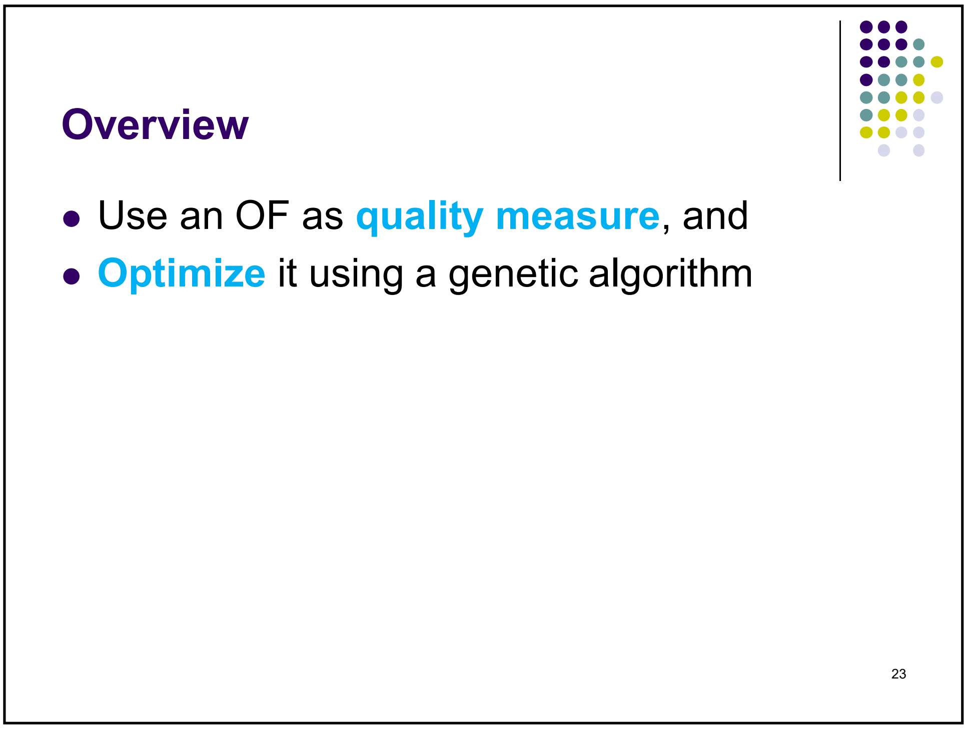
In this paper…

- SAGA: sequence alignment by genetic algorithm
  - Find **globally optimal** multiple alignments in **reasonable time**
    - As good as or better than MSA, CLUSTAL W
    - Measure: OF score, reference alignments
  - Optimize **any objective functions (OF)** one can invent
    - OF: what is the best *in real sense* → the key to success



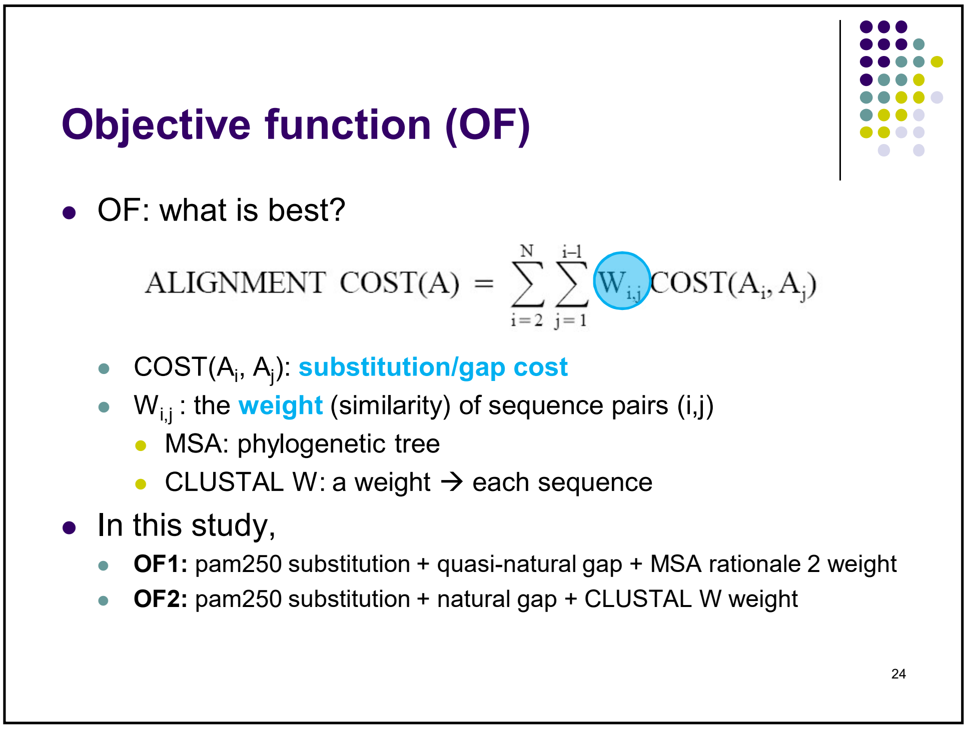
SECTION II

Methods



Overview

- Use an OF as **quality measure**, and
- **Optimize** it using a genetic algorithm



Objective function (OF)

- OF: what is best?

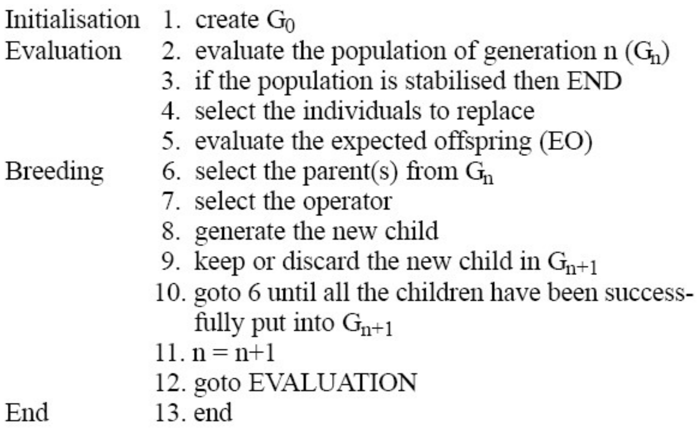
$$\text{ALIGNMENT COST(A)} = \sum_{i=2}^{N}\sum_{j=1}^{i-1} W_{i,j}\,\text{COST}(A_i, A_j)$$

  - $\text{COST}(A_i, A_j)$: **substitution/gap cost**
  - $W_{i,j}$: the **weight** (similarity) of sequence pairs (i,j)
    - MSA: phylogenetic tree
    - CLUSTAL W: a weight → each sequence
- In this study,
  - **OF1:** pam250 substitution + quasi-natural gap + MSA rationale 2 weight
  - **OF2:** pam250 substitution + natural gap + CLUSTAL W weight

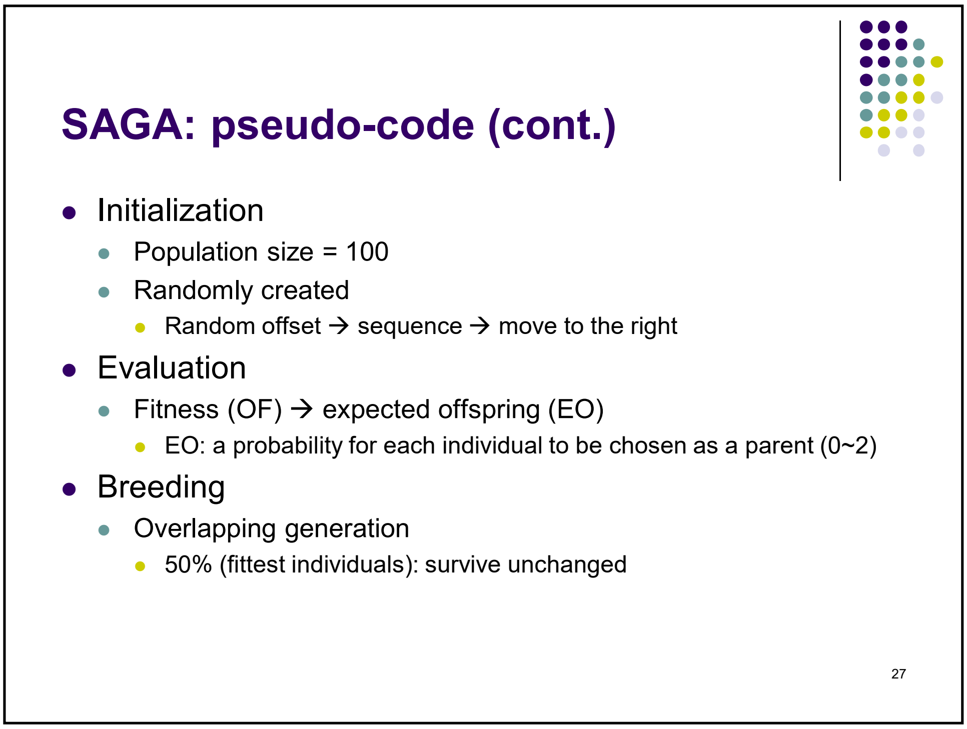
Sequence alignment by genetic algorithm (SAGA)

- Population
  - Made of alignments
- Fitness
  - Measured by the OF
- Operators
  - Each has a **probability** of being chosen → **dynamically optimized** during the run
  - Help the population to improve by **creating the children it needs**

SAGA: pseudo-code

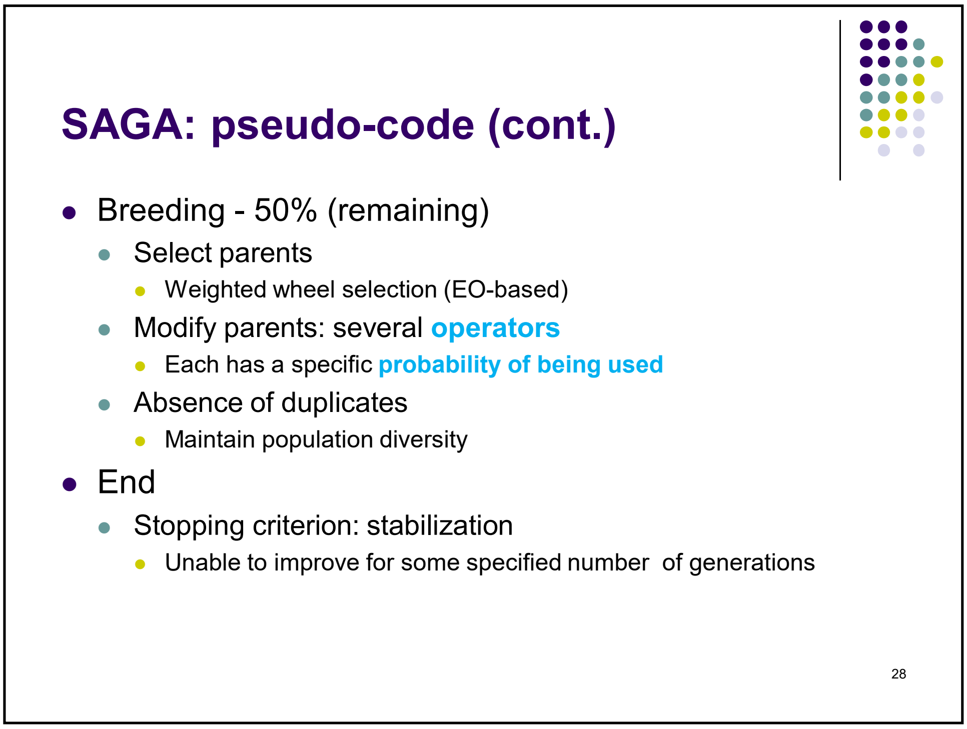


```
Initialisation  1.  create $G_0$
Evaluation      2.  evaluate the population of generation n ($G_n$)
                3.  if the population is stabilised then END
                4.  select the individuals to replace
                5.  evaluate the expected offspring (EO)
Breeding        6.  select the parent(s) from $G_n$
                7.  select the operator
                8.  generate the new child
                9.  keep or discard the new child in $G_{n+1}$
                10. goto 6 until all the children have been success-
                    fully put into $G_{n+1}$
                11. n = n+1
                12. goto EVALUATION
End             13. end
```



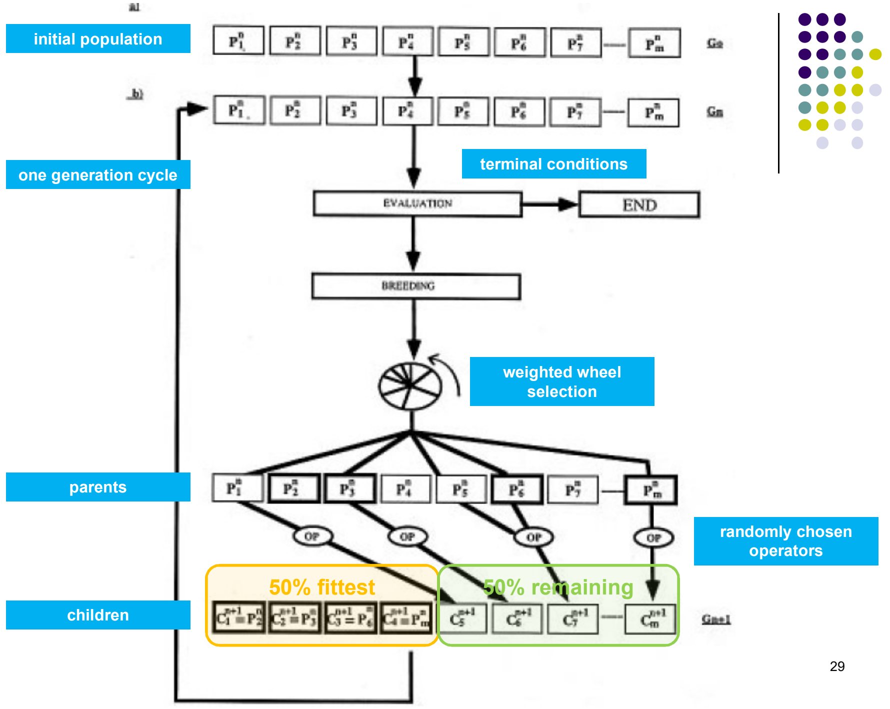
SAGA: pseudo-code (cont.)

- Initialization
  - Population size = 100
  - Randomly created
    - Random offset → sequence → move to the right
- Evaluation
  - Fitness (OF) → expected offspring (EO)
    - EO: a probability for each individual to be chosen as a parent (0~2)
- Breeding
  - Overlapping generation
    - 50% (fittest individuals): survive unchanged



SAGA: pseudo-code (cont.)

- Breeding - 50% (remaining)
  - Select parents
    - Weighted wheel selection (EO-based)
  - Modify parents: several **operators**
    - Each has a specific **probability of being used**
  - Absence of duplicates
    - Maintain population diversity
- End
  - Stopping criterion: stabilization
    - Unable to improve for some specified number of generations


a)
P_1^n P_2^n P_3^n P_4^n P_5^n P_6^n P_7^n — P_m^n
Gn
b)
P_1^n P_2^n P_3^n P_4^n P_5^n P_6^n P_7^n — P_m^n
Gn
EVALUATION
END
BREEDING
P_1^n P_2^n P_3^n P_4^n P_5^n P_6^n P_7^n — P_m^n
OP
OP
OP
OP
$C_1^{n+1}=P_2^n$ $C_2^{n+1}=P_3^n$ $C_3^{n+1}=P_6^n$ $C_4^{n+1}=P_m^n$ C_5^{n+1} C_6^{n+1} C_7^{n+1} — C_m^{n+1}
Gn+1

initial population

one generation cycle

terminal conditions

weighted wheel selection

parents

randomly chosen operators

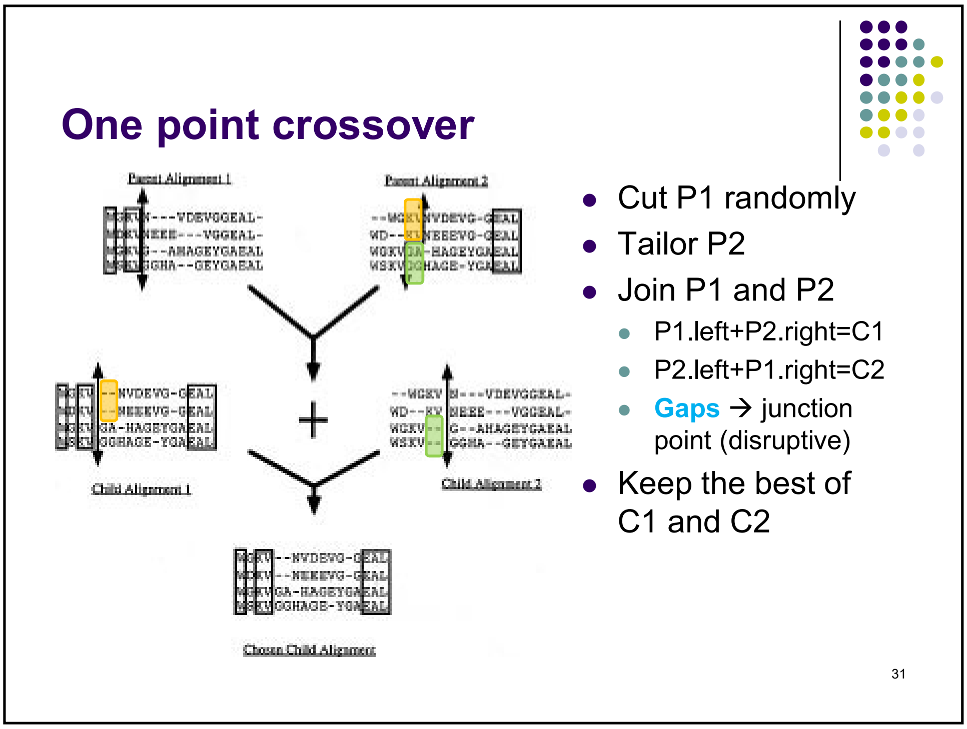
children

50% fittest

50% remaining

The operators in SAGA

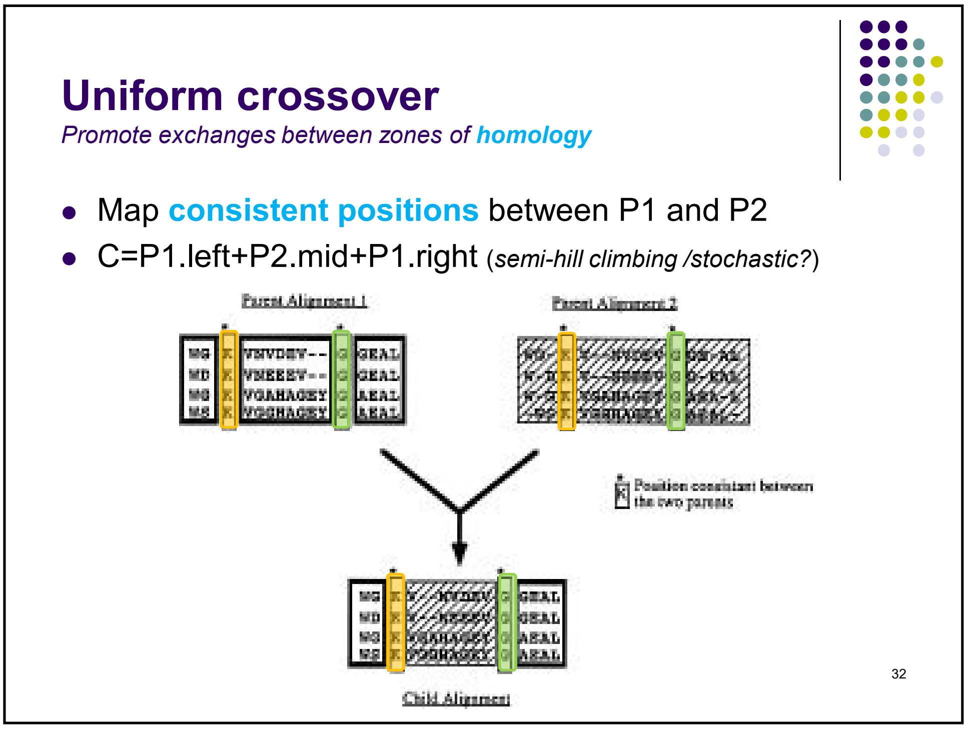
- Two types of operators
  - Crossover
    - Merge parent alignments
    - Two parents → one child
  - Mutation
    - Modifications
    - One parent → one child



One point crossover

Parent Alignment 1
Parent Alignment 2
Child Alignment 1
Child Alignment 2
Chosen Child Alignment

- Cut P1 randomly
- Tailor P2
- Join P1 and P2
  - P1.left+P2.right=C1
  - P2.left+P1.right=C2
  - **Gaps** → junction point (disruptive)
- Keep the best of C1 and C2

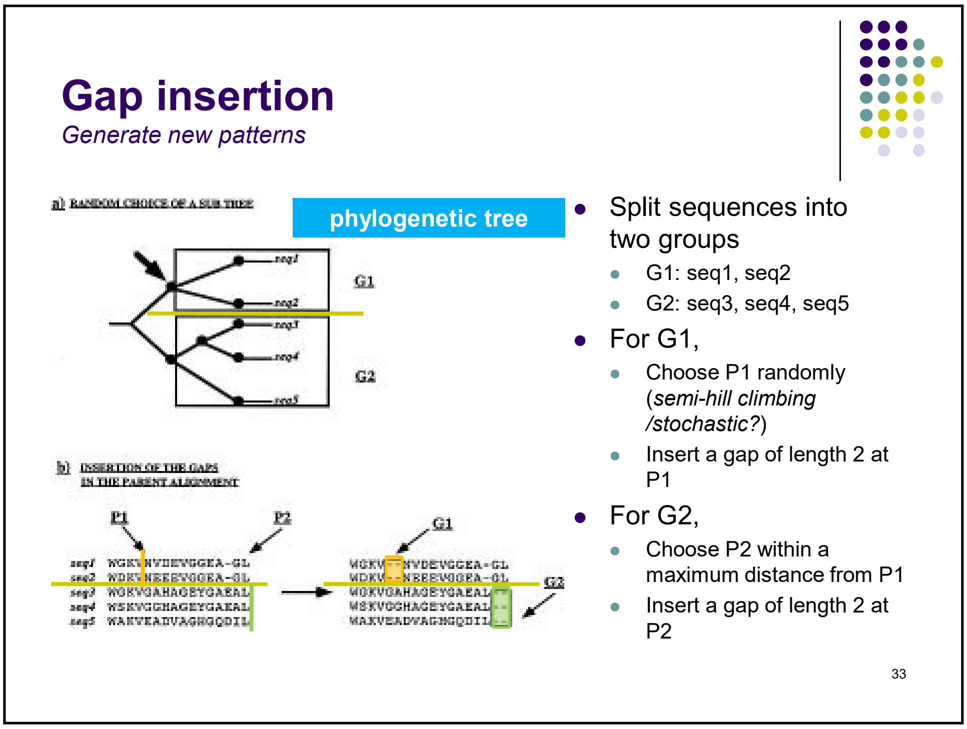


Uniform crossover

Promote exchanges between zones of ***homology***

- Map **consistent positions** between P1 and P2
- C=P1.left+P2.mid+P1.right (*semi-hill climbing /stochastic?*)

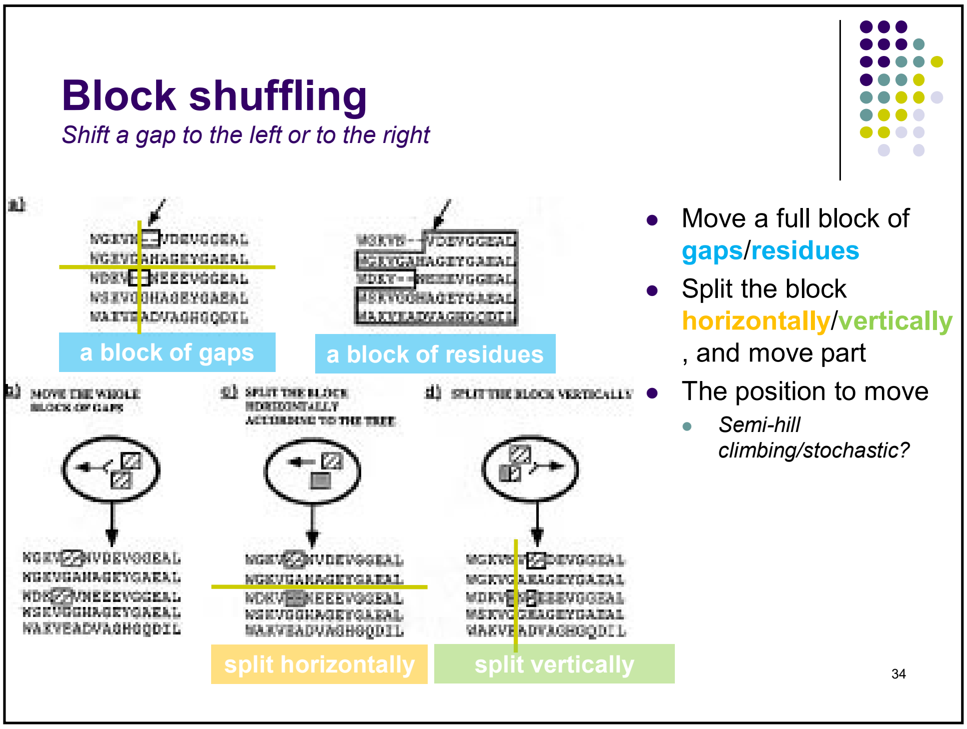
Parent Alignment 1
Parent Alignment 2
Position consistant between the two parents
Child Alignment



Gap insertion

Generate new patterns

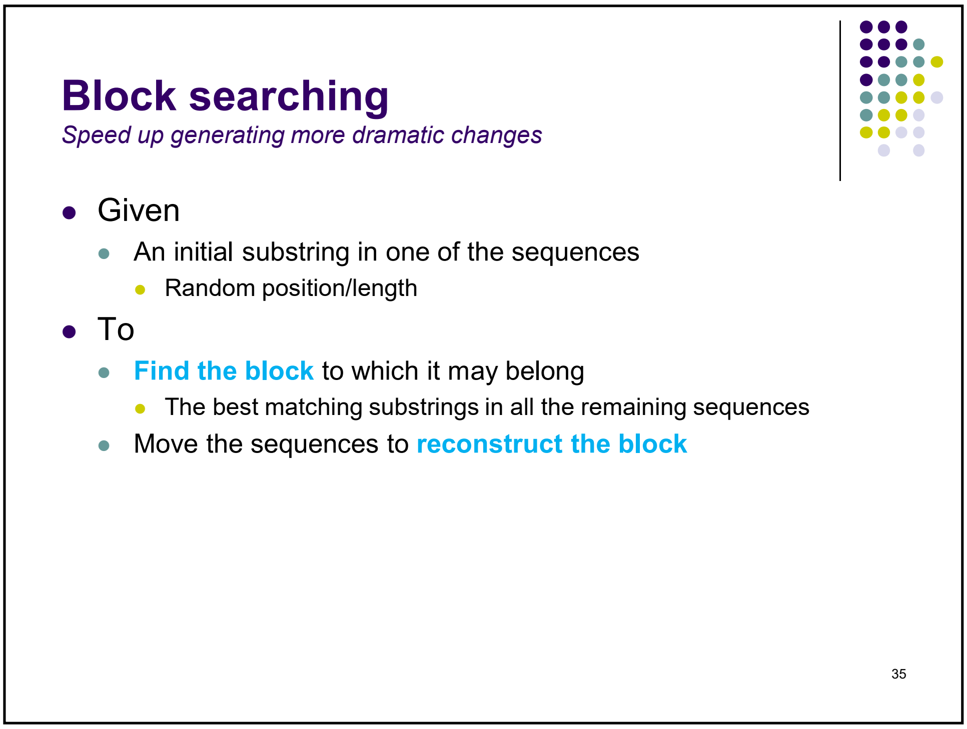
- Split sequences into two groups
  - G1: seq1, seq2
  - G2: seq3, seq4, seq5
- For G1,
  - Choose P1 randomly (*semi-hill climbing /stochastic?*)
  - Insert a gap of length 2 at P1
- For G2,
  - Choose P2 within a maximum distance from P1
  - Insert a gap of length 2 at P2



Block shuffling

Shift a gap to the left or to the right

- Move a full block of gaps/residues
- Split the block horizontally/vertically, and move part
- The position to move
  - *Semi-hill climbing/stochastic?*

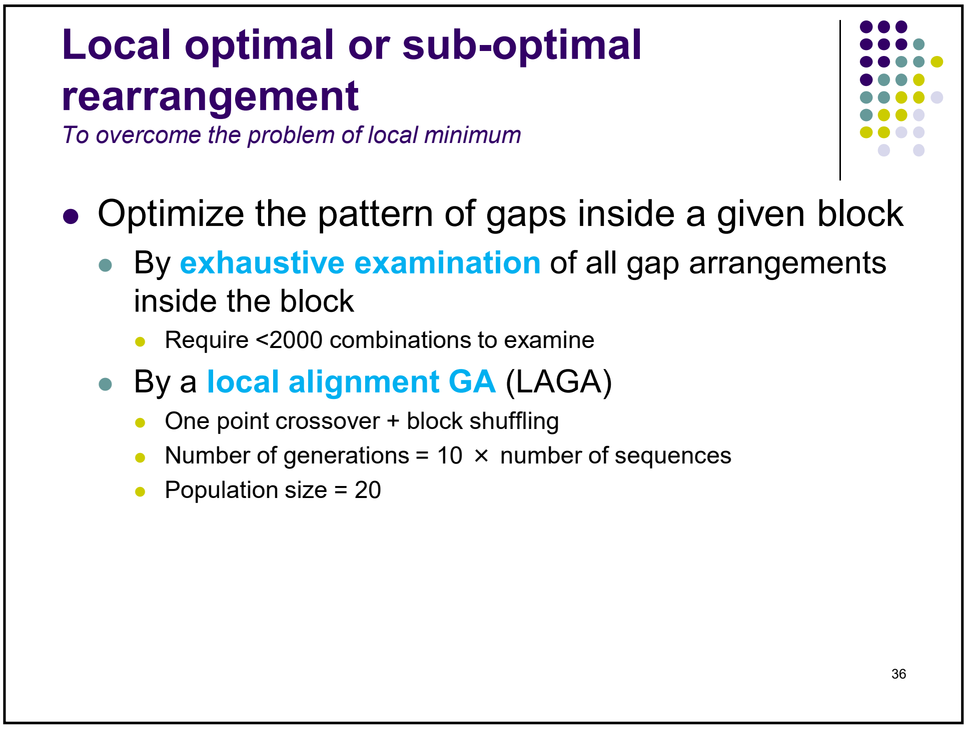


Block searching

Speed up generating more dramatic changes

- Given
  - An initial substring in one of the sequences
    - Random position/length
- To
  - **Find the block** to which it may belong
    - The best matching substrings in all the remaining sequences
  - Move the sequences to **reconstruct the block**

35

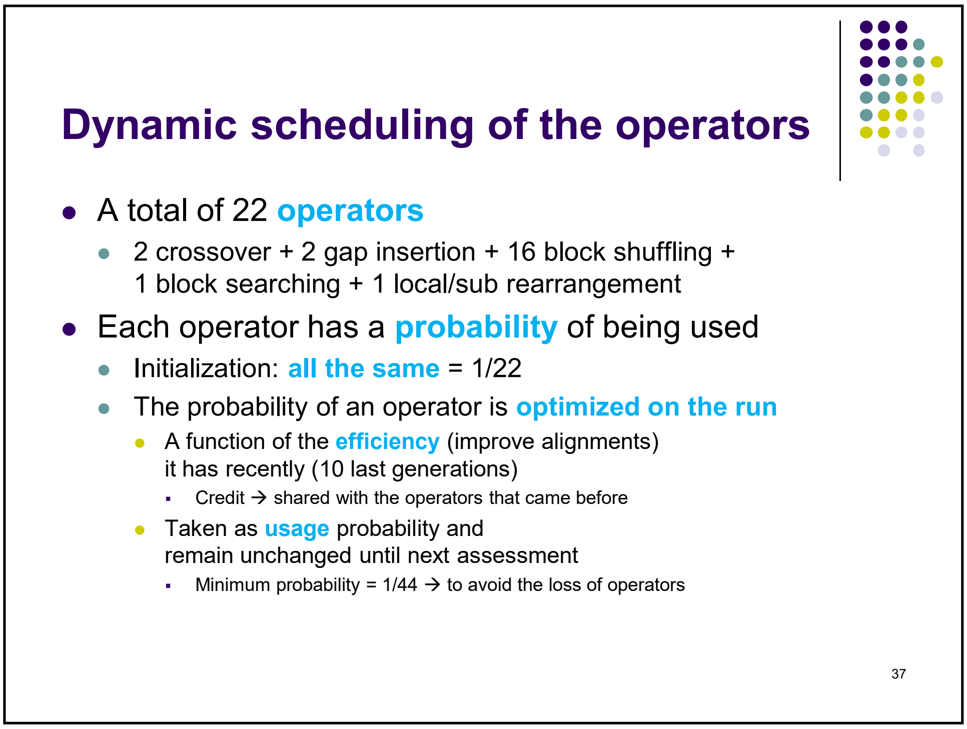


Local optimal or sub-optimal rearrangement

To overcome the problem of local minimum

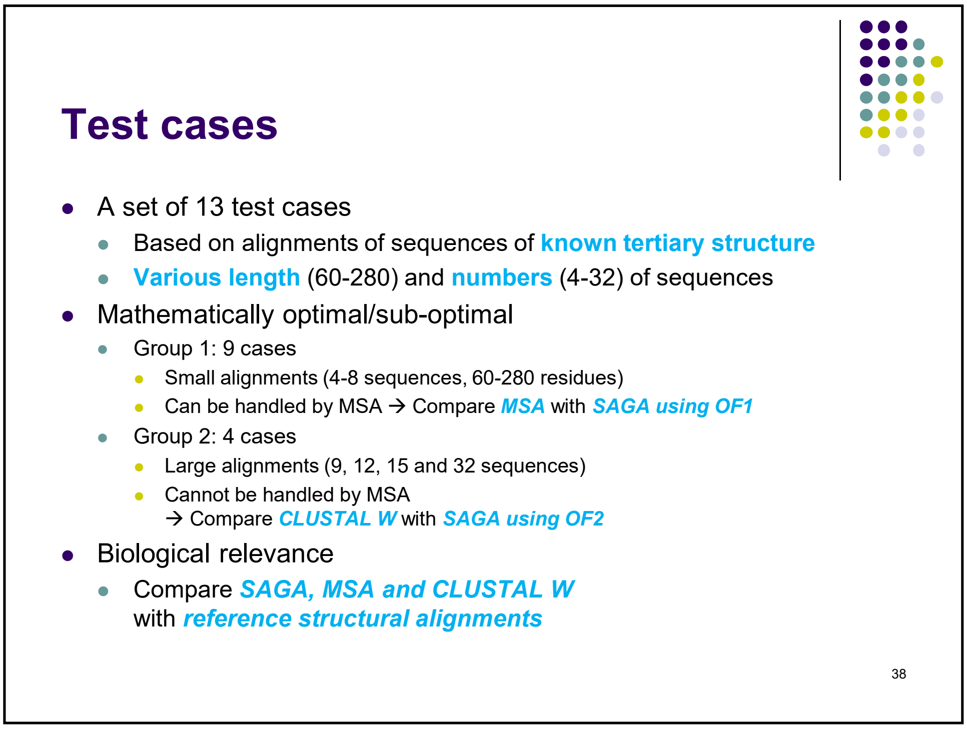
- Optimize the pattern of gaps inside a given block
  - By **exhaustive examination** of all gap arrangements inside the block
    - Require <2000 combinations to examine
  - By a **local alignment GA** (LAGA)
    - One point crossover + block shuffling
    - Number of generations = 10 × number of sequences
    - Population size = 20

36



Dynamic scheduling of the operators

- A total of 22 **operators**
  - 2 crossover + 2 gap insertion + 16 block shuffling + 1 block searching + 1 local/sub rearrangement
- Each operator has a **probability** of being used
  - Initialization: **all the same** = 1/22
  - The probability of an operator is **optimized on the run**
    - A function of the **efficiency** (improve alignments) it has recently (10 last generations)
      - Credit → shared with the operators that came before
    - Taken as **usage** probability and remain unchanged until next assessment
      - Minimum probability = 1/44 → to avoid the loss of operators



Test cases

- A set of 13 test cases
  - Based on alignments of sequences of **known tertiary structure**
  - **Various length** (60-280) and **numbers** (4-32) of sequences
- Mathematically optimal/sub-optimal
  - Group 1: 9 cases
    - Small alignments (4-8 sequences, 60-280 residues)
    - Can be handled by MSA → Compare ***MSA*** with ***SAGA using OF1***
  - Group 2: 4 cases
    - Large alignments (9, 12, 15 and 32 sequences)
    - Cannot be handled by MSA
      → Compare ***CLUSTAL W*** with ***SAGA using OF2***
- Biological relevance
  - Compare ***SAGA, MSA and CLUSTAL W*** with ***reference structural alignments***

SECTION III

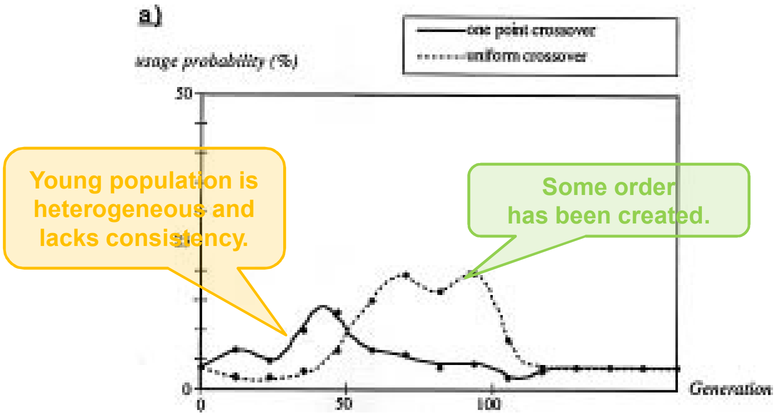
Results

Self tuning ability (1)

Automatic scheduling of the operators

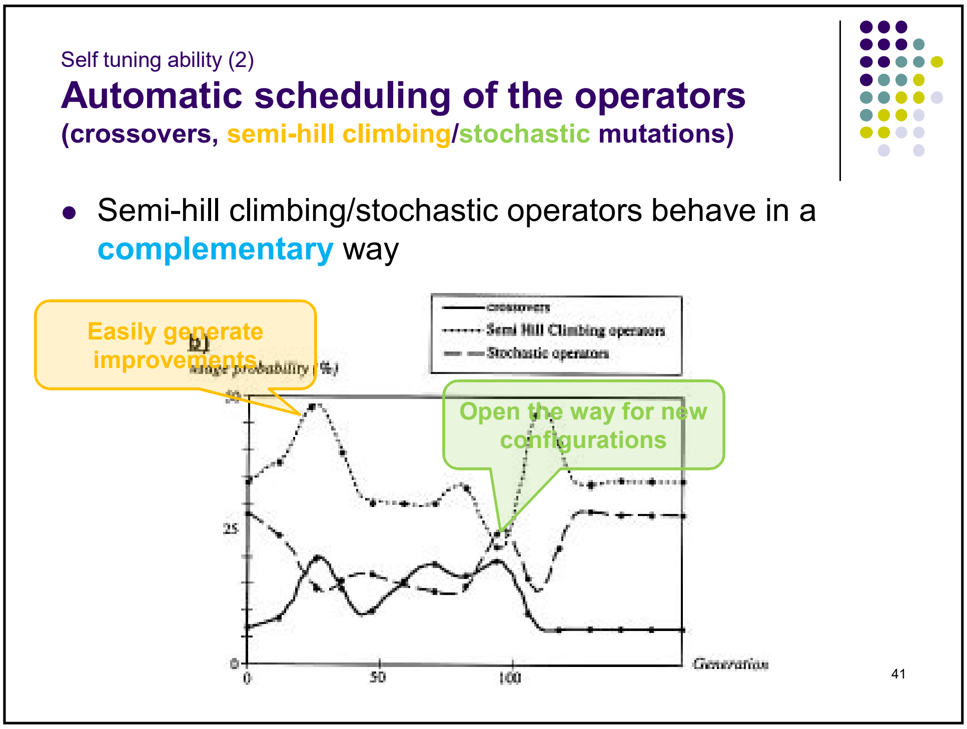
(one point/uniform crossover)

- Two types of crossover are **competing** with each other



Young population is heterogeneous and lacks consistency.

Some order has been created.



Self tuning ability (2)

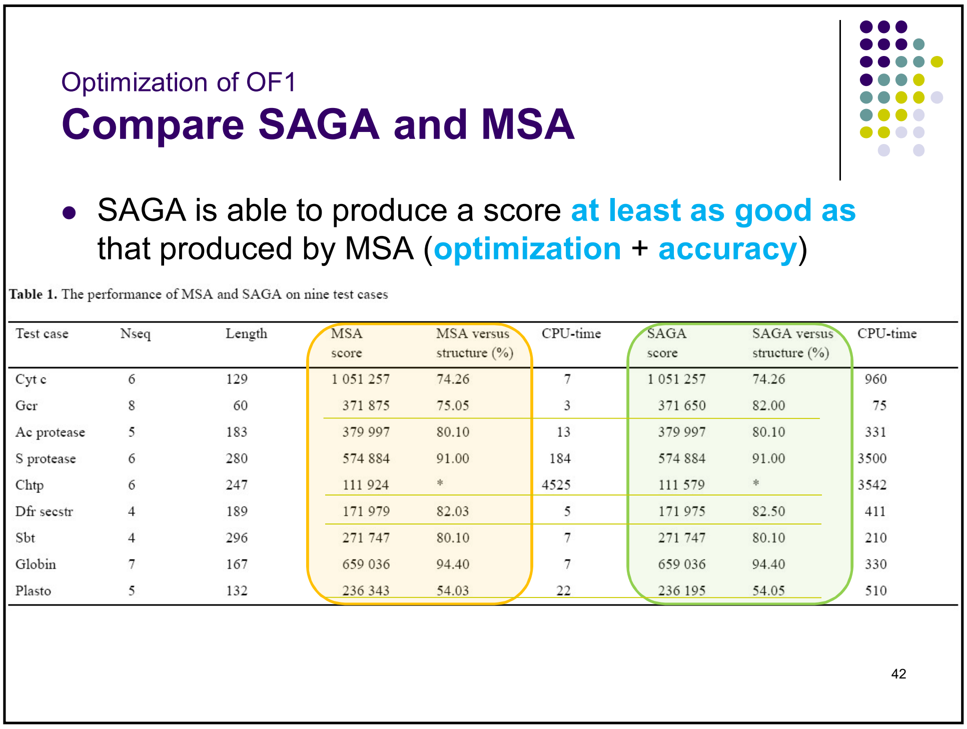
Automatic scheduling of the operators

(crossovers, semi-hill climbing/stochastic mutations)

- Semi-hill climbing/stochastic operators behave in a **complementary** way

Easily generate improvements

Open the way for new configurations



Optimization of OF1

Compare SAGA and MSA

- SAGA is able to produce a score **at least as good as** that produced by MSA (**optimization** + **accuracy**)

Table 1. The performance of MSA and SAGA on nine test cases

Test case	Nseq	Length	MSA score	MSA versus structure (%)	CPU-time	SAGA score	SAGA versus structure (%)	CPU-time
Cyt c	6	129	1 051 257	74.26	7	1 051 257	74.26	960
Gcr	8	60	371 875	75.05	3	371 650	82.00	75
Ac protease	5	183	379 997	80.10	13	379 997	80.10	331
S protease	6	280	574 884	91.00	184	574 884	91.00	3500
Chtp	6	247	111 924	*	4525	111 579	*	3542
Dfr secstr	4	189	171 979	82.03	5	171 975	82.50	411
Sbt	4	296	271 747	80.10	7	271 747	80.10	210
Globin	7	167	659 036	94.40	7	659 036	94.40	330
Plasto	5	132	236 343	54.03	22	236 195	54.05	510

Optimization of OF2 (1)

Compare SAGA and CLUSTAL W

- SAGA performs **more accurately** than CLUSTAL W on data sets of **realistic size**

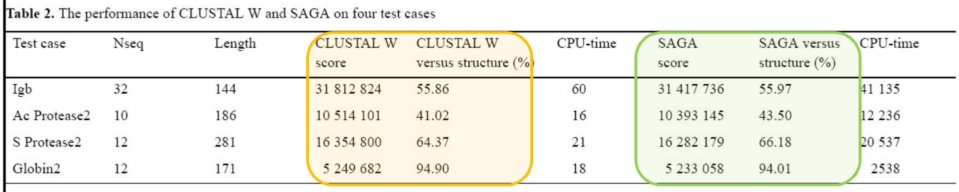
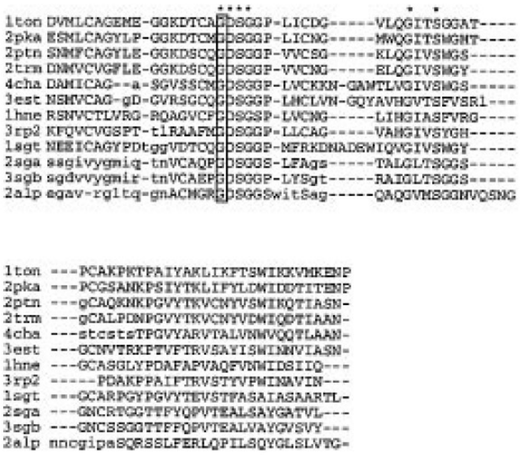


Table 2. The performance of CLUSTAL W and SAGA on four test cases

Test case	Nseq	Length	CLUSTAL W score	CLUSTAL W versus structure (%)	CPU-time	SAGA score	SAGA versus structure (%)	CPU-time
Igb	32	144	31 812 824	55.86	60	31 417 736	55.97	41 135
Ac Protease2	10	186	10 514 101	41.02	16	10 393 145	43.50	12 236
S Protease2	12	281	16 354 800	64.37	21	16 282 179	66.18	20 537
Globin2	12	171	5 249 682	94.90	18	5 233 058	94.01	2538

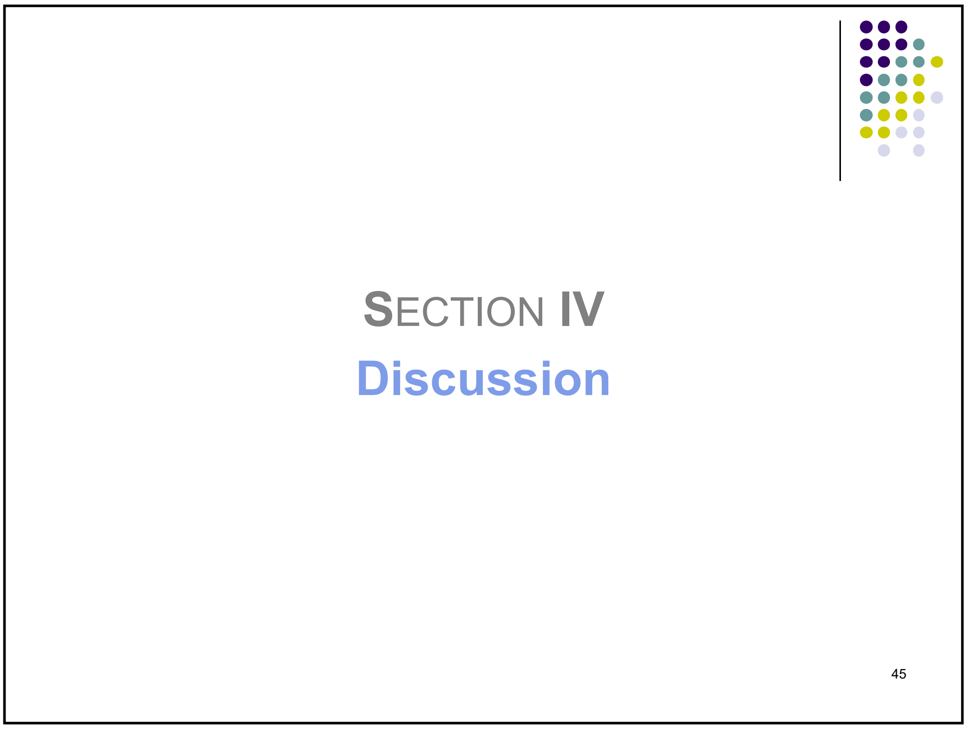
Optimization of OF2 (2)

Compare SAGA and CLUSTAL W

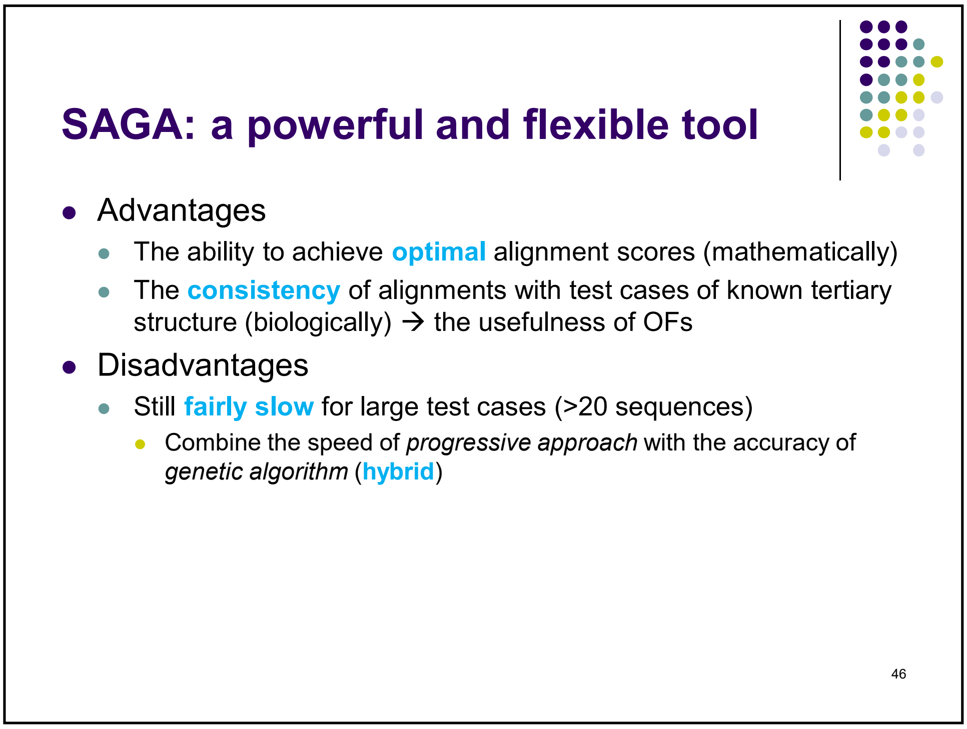


```
                                  ****                  *  *
1ton DVMLCAGEME-GGKDTCAGDSGGP-LICDG-----VLQGITSGGAT----
2pka ESMLCAGYLP-GGKDTCMGDSGGP-LICNG-----MWQGITSWGHT----
2ptn SNMFCAGYLE-GGKDSCQGDSGGP-VVCSG-----KLQGIVSWGS-----
2trm DNMVCVGFLE-GGKDSCQGDSGGP-VVCNG-----ELQGIVSWGY-----
4cha DAMICAG--a-SGVSSCMGDSGGP-LVCKKN-GAWTLVGIVSWGS-----
3est NSMVCAG-gD-GVRSGCQGDSGGP-LHCLVN-GQYAVHGVTSFVSRl---
1hne RSNVCTLVRG-RQAGVCFGDSGSP-LVCNG-----LIHGIASFVRG----
3rp2 KFQVCVGSPT-tlRAAFMGDSGGP-LLCAG-----VAHGIVSYGH-----
1sgt NEEICAGYPDtggVDTCQGDSGGP-MFRKDNADEWIQVGIVSWGY-----
2sga asgivygmiq-tnVCAQPGDSGGS-LFAgs-----TALGLTSGGS-----
3sgb sgdvvygmir-tnVCAEPGDSGGP-LYSgt-----RAIGLTSGGS-----
2alp egav-rgltq-gnACMGRGDSGGSwitSag-----QAQGVMSGGNVQSNG

1ton ---PCAKPKTPAIYAKLIKFTSWIKKVMKENP
2pka ---PCGSANKPSIYTKLIFYLDWIDDTITENP
2ptn ---gCAQKNKPGVYTKVCNYVSWIKQTIASN-
2trm ---gCALPDNPGVYTKVCNYVDWIQDTIAAN-
4cha ---stcstsTPGVYARVTALVNWVQQTLAAN-
3est ---GCNVTRKPTVFTRVSAYISWINNVIASN-
1hne ---GCASGLYPDAFAPVAQFVNWIDSIIQ---
3rp2 -----PDAKPPAIFTRVSTYVPWINAVIN---
1sgt ---GCARPGYPGVYTEVSTFASAIASAARTL-
2sga ---GNCRTGGTTFYQPVTEALSAYGATVL---
3sgb ---GNCSSGGTTFFQPVTEALVAYGVSVY---
2alp mncgipaSQRSSLFERLQPILSQYGLSLVTG-
```

- SAGA accurately finds the main features
  - 12 completely conserved positions
    - SAGA: 11
    - CLUSTAL W: 10



SECTION IV
Discussion



SAGA: a powerful and flexible tool

- Advantages
  - The ability to achieve **optimal** alignment scores (mathematically)
  - The **consistency** of alignments with test cases of known tertiary structure (biologically) → the usefulness of OFs
- Disadvantages
  - Still **fairly slow** for large test cases (>20 sequences)
    - Combine the speed of *progressive approach* with the accuracy of *genetic algorithm* (**hybrid**)

Starting population

- Currently, seed alignments completely **randomly**
  - Use **heuristic** alignments generated by CLUSTAL W
  - Could be trapped in local minima
- SAGA as an alignment **improver**
  - Starting alignment → close to the optimal solution
  - Generate hybrid alignments for **very large test cases**

Efficiency of GA

- Use a large number of mutational and crossover **operators**, and automatically **schedule** them
  - Complicated and cumbersome?
    - MSA is **not a simple problem**
  - The most useful operators
    - Based on **biological reality**
      (e.g. moving blocks using the tree as a guide)
  - Automatic scheduling
    - New situation/problem → **new operators**
    - **Usefulness or redundancy** at different stages
- Implement and test any **OF** one can think of
  - A good **measure of quality** → key to success

Questions after sequence alignment

- Q1: Is the alignment **significant** with respect to some statistical model?
  - A very difficult problem which has solutions for two sequences under certain conditions
- Q2: How **stable** is the alignment or which pieces of the alignment are stable?
  - Important to interpret new alignments and there are solutions for just two sequences

The End

Thank you

for your attention!