@ BY KUANGC

BLAST Algorithms

Basic Local Alignment Search Tool

指导老师：罗静初（北京大学）
汇报人　：匡　琛
地址　　：北京市海淀区中关村南大街12号中国农业科学院
邮箱　　：junyuankuang@icloud.com

(2016.11)

概要OUTLINE

1、背景介绍BACKGROUND ▶ 是什么

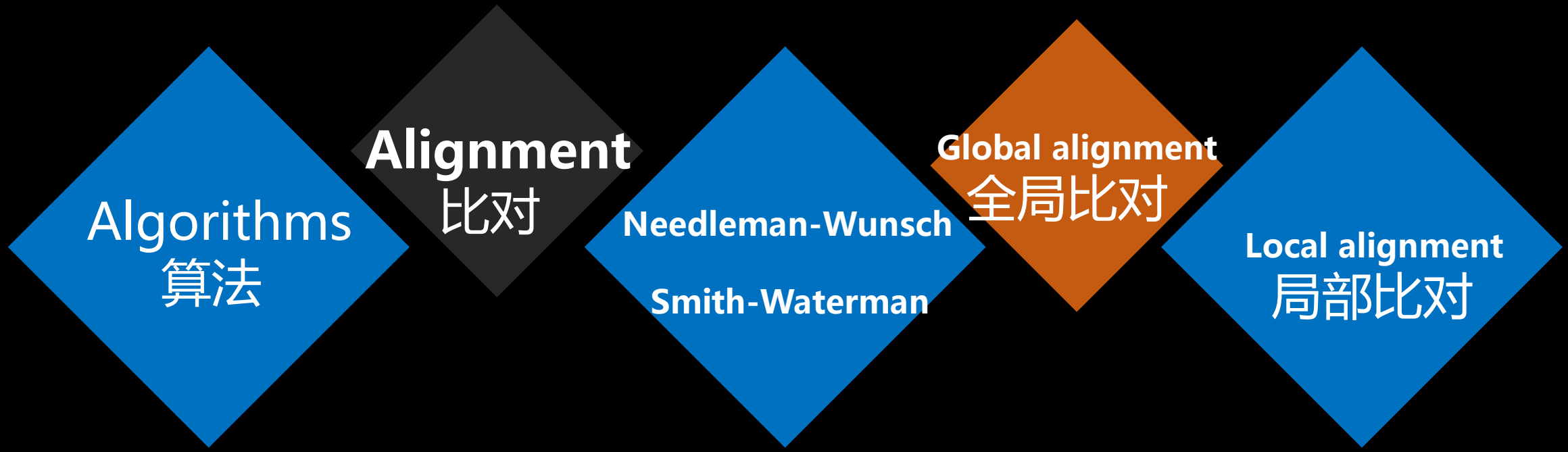
2、算法基础ANALYSIS ▶ 怎么做

3、应用讨论DISCUSSION ▶ 为什么

PART ONE / 背景介绍

PART ONE 1

基本概念KEY WORDS



PART ONE 1

基本概念KEY WORDS

Query
搜索序列

Databases
数据库

Matrix
矩阵
Scoring matrix
打分矩阵

Gap
空位
Gap existence
产生一个空位
Gap extension
空位延伸

Rigorous algorithm
严格算法

PART ONE 1

基本概念KEY WORDS

Heuristic algorithm
启发式算法

Score (S)
得分

E-value
E值
P-value
p值

seed
种子序列

high scoring segment pair
高得分片段HSP

PART ONE 2

算法Algorithms

定义：解题方案的准确而完整的描述，是解决问题的一系列清晰指令。

举个例子：如何把大象装到冰箱里？

1、把冰箱门打开

2、把大象装进去

3、把冰箱门关上

PART ONE 2

算法Algorithms

定义：解题方案的准确而完整的描述，是解决问题的一系列清晰指令。

再举个例子：引入数学排序问题的伪代码来展示算法

***Input（输入）*: 数列** $< a_1, a_2, a_3 \ldots, a_n >$

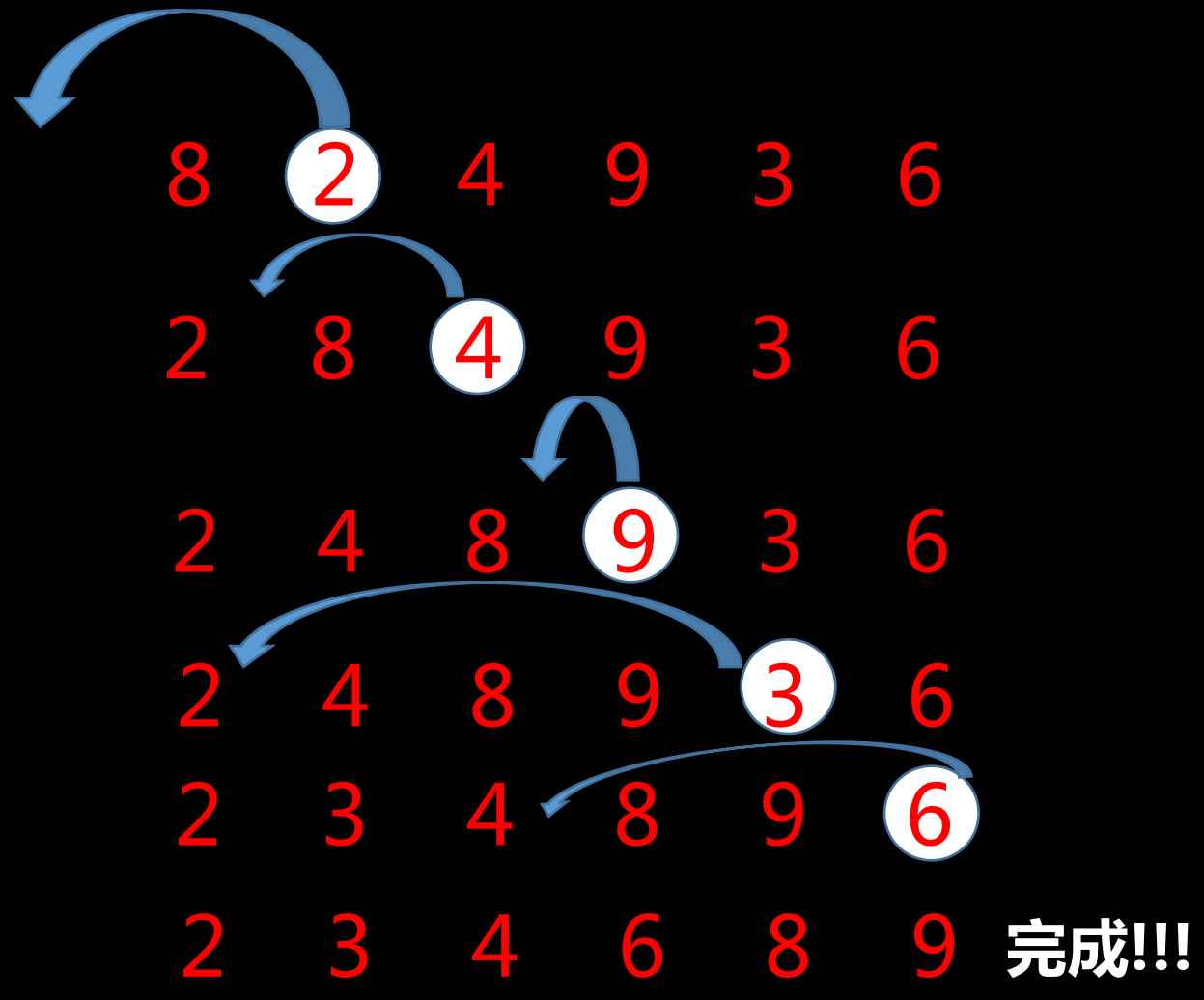
***Output（输出）*: 数列** $< a_1', a_2', a_3' \ldots a_n' >$

Example（例子）*:** ***Input : **8 2 4 9 3 6**

Output : **2 3 4 6 8 9**

PART ONE 2

算法Algorithms



PART ONE 2

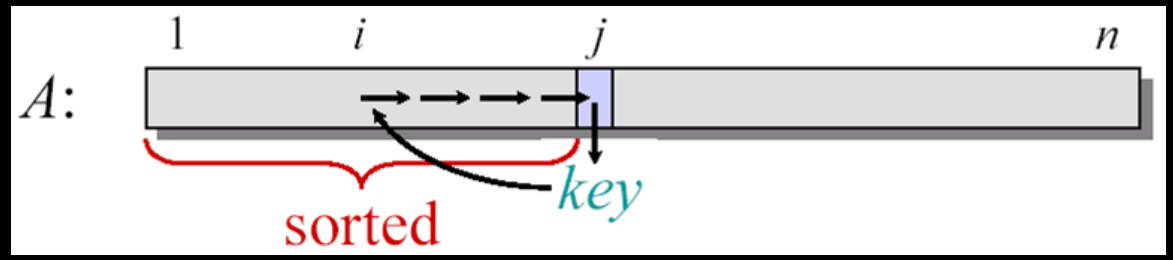
算法Algorithms

定义：解题方案的准确而完整的描述，是解决问题的一系列清晰指令。

伪代码：

插入排序（A，n）即：A[1,2....n]

```
For j ⟵ 2 to n
do key ⟵ A[j]
   i ⟵ j-1
   While i > 0 and A[i] > key
   Do A[i+1] ⟵ A[i]
      i ⟵ i-1
   A[i+1] = key
```



PART ONE 3

比对算法起源ORIGIN

Needleman-Wunsch, 1970

- Global alignment 全局比对
- Rigorous algorithm 严格算法
- -Dynamic programming 动态规划
- -Simple to implement 操作容易
- Slow; not used for search

I

Smith-Waterman, 1981

- Local alignment 局部比对
- Rigorous algorithm 严格算法
- -Dynamic programming 动态规划
- -Fairly simple to implement 操作相当容易
- Precise, sensitive alignments
- Slow; not used for search

II

发展历程

FASTA, Lipman Pearson 1985

- Local alignment 局部比对
- Heuristic algorithm 启发式算法
- Faster than Smith-Waterman

III

IV

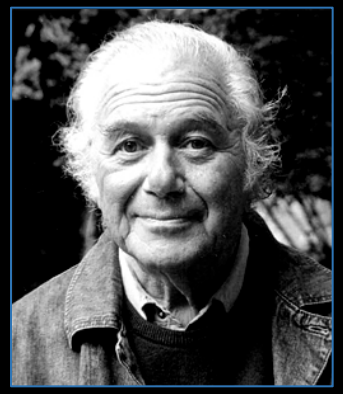
BLAST, Altschul et al 1990

- Basic Local Alignment Search Tool
- Heuristic algorithm
- ~50 faster than Smith-Waterman
- Faster than FASTA, less sensitive 敏感度较低

PART ONE 4

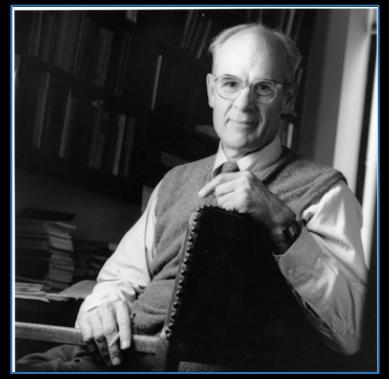
算法科学家Scientists

Saul B. Needleman



Christian D. Wunsch

Peter H. Sellers



Temple F. Smith

Michael Waterman

PART ONE 4

算法科学家Scientists

David J. Lipman

William R. Pearson

Stephen Frank Altschul

PART TWO

算法基础

PART TWO 1

序列比对Alignment

- 相似的序列 → 相似的结构 → 相似的功能

  推断这个未知新序列的可能的功能

- 相似的序列 → 同源

  在演化分析中用来构建演化树的重要依据

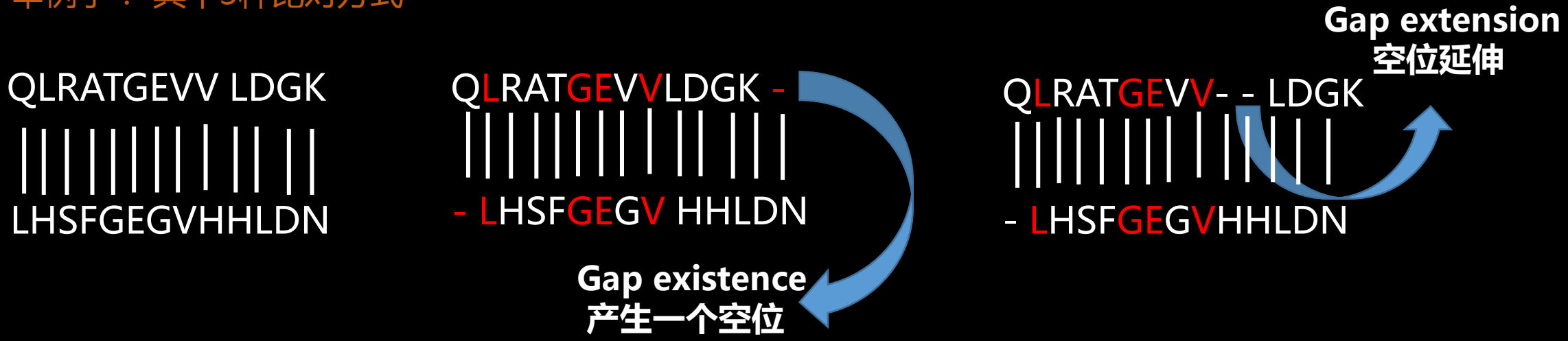
PART TWO 2

双序列比对2-D Alignment

蚕豆血红蛋白 QLRATGEVVLDGK

马β珠蛋白 LHSFGEGVHHLDN

举例子： 其中3种比对方式



PART TWO 2

常用打分矩阵 Scoring matrix

BLOSUM62

Block Substitution Matrix

块替换矩阵

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

https://en.wikipedia.org/wiki/BLOSUM

PART TWO 2

常用打分矩阵 Scoring matrix

PAM 250

point accepted mutation

点接受突变

PAM 250

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

http://image.baidu.com/search/detail

PART TWO 2

双序列比对2-D Alignment

QLRATGEVV LDGK

||||||||| ||| ||

LHSFGEGVHHLDN

-2 -3 -1 -2 -2 -2 -2 4 -3 -3 -4 -1 0

-21

定义：Gap existence: -11；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

QLRATGEVVLDGK -

- LHSFGEGV HHLDN

-11
4
0
1
-2
6
5
-3
4
-3
-1
-4
-1
-11

-16

定义：Gap existence: -11；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

QLRATGEVV- - LDGK

- LHSFGEGVHHLDN

-11 4 0 1 -2 6 5 -3 4 -11 -1 4 6 0 -11

-9

定义：Gap existence: -11；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

定义：Gap existence: -11；Gap extension: -1

采用**BLOSUM62打分矩阵：**

QLRATGEVV LDGK

LHSFGEGVHHLDN

-21

QLRATGEVVLDGK -

- LHSFGEGV HHLDN

-16

QLRATGEVV- - LDGK

- LHSFGEGVHHLDN

-9

PART TWO 2

双序列比对2-D Alignment

QLRATGEVV LDGK

||||||||| | || ||

LHSFGEGVHHLDN

-21

定义：Gap existence: -15；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

QLRATGEVVLDGK -

- LHSFGEGV HHLDN

-24

定义：Gap existence: -15；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

```
QLRATGEVV- - LDGK
||||||||| |||||
- LHSFGEGVHHLDN
```

-21

定义：Gap existence: -15；Gap extension: -1

采用BLOSUM62打分矩阵：

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

PART TWO 2

双序列比对2-D Alignment

定义：Gap existence: -15；Gap extension: -1

采用**BLOSUM62打分矩阵：**

QLRATGEVV LDGK

LHSFGEGVHHLDN

-21

QLRATGEVVLDGK -

- LHSFGEGV HHLDN

-24

QLRATGEVV- - LDGK

- LHSFGEGVHHLDN

-21

PART TWO 2

双序列比对2-D Alignment

QLRATGEVV LDGK

LHSFGEGVHHLDN

① -21

② -21

QLRATGEVVLDGK -

- LHSFGEGV HHLDN

-16

-24

QLRATGEVV- - LDGK

- LHSFGEGVHHLDN

-9

-21

小结：

改变空位罚分即改变最佳比对结果

得分不是衡量比对优劣的决定因素

PART TWO 3

双序列比对类型Type

1

全局比对

global alignment

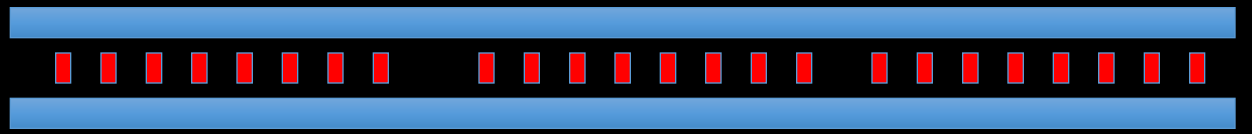
Needleman-Wunsch

全局比对=全部比对

将两条序列头对头、尾对尾的比对，

关注整体的相似度 。

[needle in EMBOSS package]



PART TWO 3

双序列比对类型Type

2

局部比对

local alignment

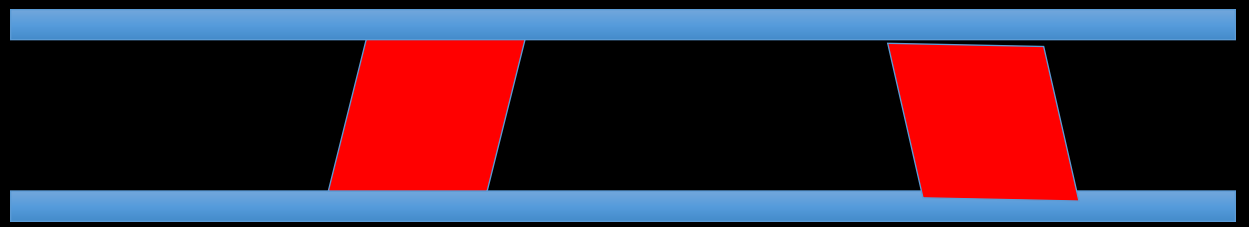
Smith-Waterman

局部比对=部分比对

将两条序列的任意部分进行比对，

关注局部的相似度

[Water in EMBOSS package]



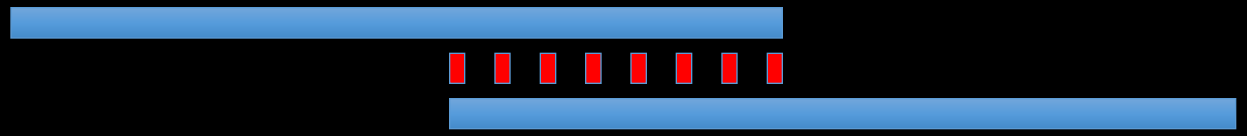
PART TWO 3

双序列比对类型Type

3

半全局比对

Semi-Global alignment



PART TWO 4

双序列比对算法 2-D Alignment Algorithms

- **双序列比对算法是所有序列比对算法的基础与核心。**
- **重要前提：位置独立性，每条序列的每个位置上的碱基都是独立的，无前后依赖性。**
- **目标是将两条序列以最相似的形式贴在一起，允许错配【mismatch】,和空位【gap】**

举例：

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

1

穷举法

●问题：对于长度分别为N和N的两条序列，总共有多少种比对形式？

●答案：一个独立的比对，相当于从N+N个位置中抽取N个位置。

因此，这个问题可转变为数学组合问题，总共有 C_{2N}^{N} 种比对

举例：AGC与ACC

A G C - - -

- - - A C C

C_6^3

●找出最佳【optimal】比对结果：遍历所有比对形式，计算每种比对形式的得分，得分最高的比对形式就是最佳比对结果。

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

1

穷举法

不足：如果需要比对的序列长，那么用穷举法吃得消吗？

举个例子：

假设一、比对序列长度为10，穷举法：C_{20}^{10} =184，756种

假设二、比对序列长度为300，穷举法：C_{600}^{300} = $7X10^{88}$ 种

结论：吃不消。

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2

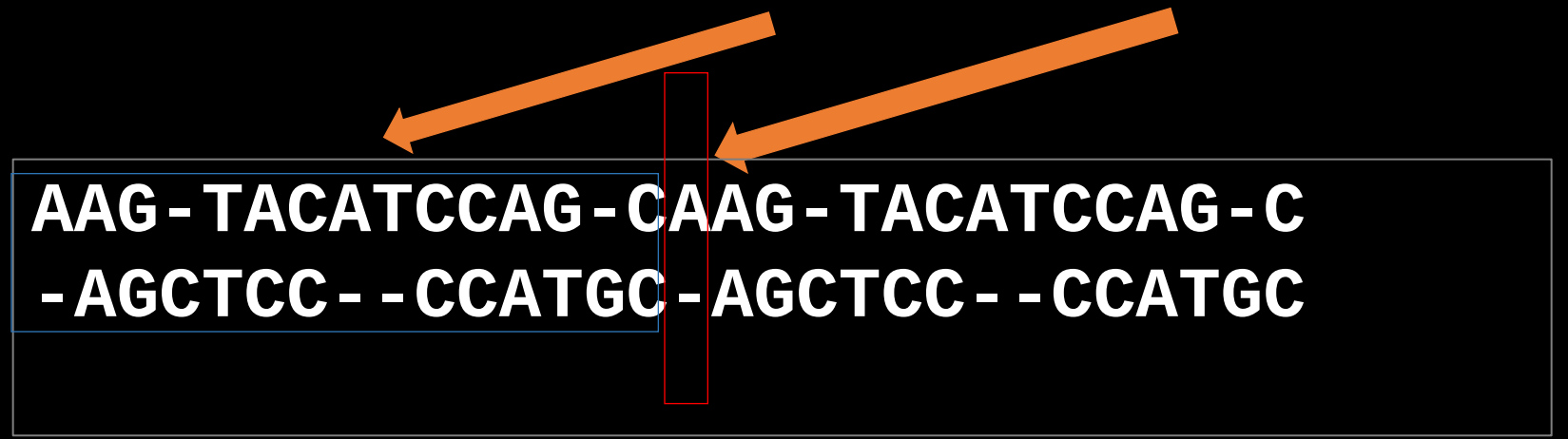
动态规划法

Dynamic programming

（严格算法）

动态规划算法：可以用最短的时间找出最佳的比对结果。

核心思想：最好的比对 = 之前最好的比对+当前最好的比对



PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中

动态规划法

Needleman-Wunsch

举个例子：比对ATCCG与AGTCG

定义：空位罚分：-2；比对上：+5，未比对：-1.

规则：

1.空位对空位：无，即填0；

2.三个方向路径

往右延伸GAP+空位罚分

往下延伸GAP+空位罚分

对角线两两比对+打分矩阵；

3.最优路径即从右下方最后一个位置回溯到空位对空位左上方值为0的整条路径。

	GAP	A	T	C	C	G
GAP	0	-2	-4	-6	-8	-10
A	-2	-4				
G	-4					
T	-6					
C	-8					
G	-10					

表示：
-
|
A

得分：
(-2)+(-2)=-4

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中

动态规划法

Needleman-Wunsch

举个例子：比对ATCCG与AGTCG

定义：空位罚分：-2；比对上：+5，未比对：-1.

规则：

1.空位对空位：无，即填0；

2.三个方向路径

往右延伸GAP+空位罚分

往下延伸GAP+空位罚分

对角线两两比对+打分矩阵；

3.最优路径即从右下方最后一个位置回溯到空位对空位左上方值为0的整条路径。

	GAP	A	T	C	C	G
GAP	0	-2	-4	-6	-8	-10
A	-2	-4				
G	-4					
T	-6					
C	-8					
G	-10					

表示：A
-

得分：
（-2）+（-2）=-4

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中

动态规划法

Needleman-Wunsch

举个例子：比对ATCCG与AGTCG

定义：空位罚分：-2；比对上：+5，未比对：-1.

规则：

1.空位对空位：无，即填0；

2.三个方向路径

往右延伸GAP+空位罚分

往下延伸GAP+空位罚分

对角线两两比对+打分矩阵；

3.最优路径即从右下方最后一个位置回溯到空位对空位左上方值为0的整条路径。

	GAP	A	T	C	C	G
GAP	0	-2	-4	-6	-8	-10
A	-2					
G	-4					
T	-6					
C	-8					
G	-10					

-4

-4

5

表示：A

A

得分：

（0）+（5）=5

结论：

取三个方向中最大值5

并保留对角线方向。

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

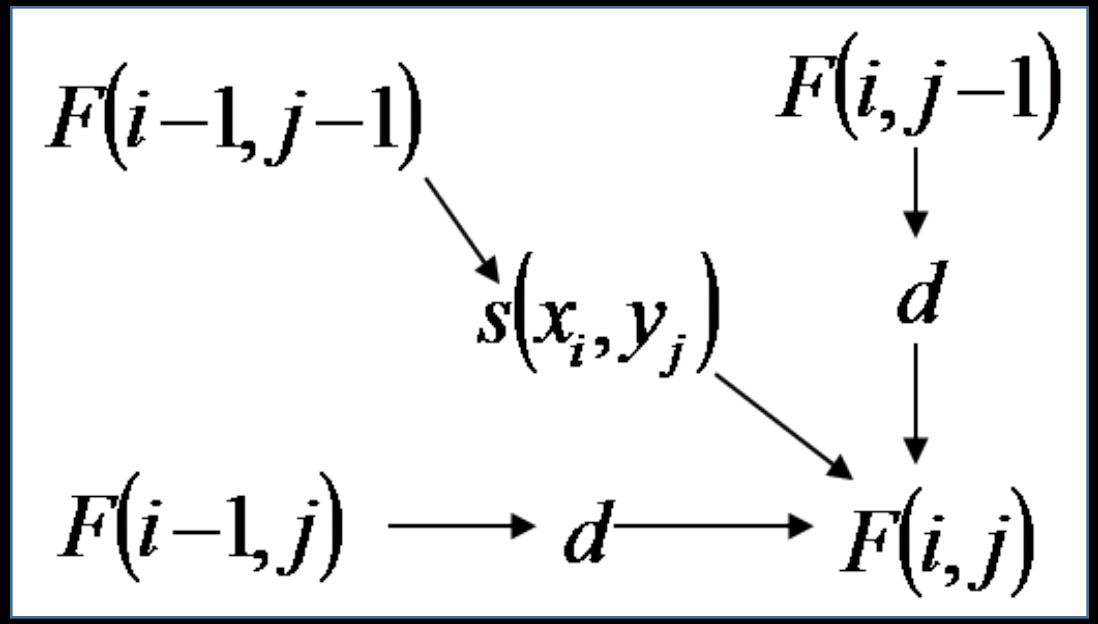
2.1

全局比对中

动态规划法

Needleman-Wunsch

全局比对动态算法：



取三个方向的最大值
并保留这个方向。

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中

动态规划法

Needleman-Wunsch

按照动态规划算法填满表格：

	GAP	A	T	C	C	G
GAP	0	-2	-4	-6	-8	-10
A	-2	5	3	1	-1	-3
G	-4	3	4	2	0	4
T	-6	1	8	6	4	2
C	-8	-1	6	13	11	9
G	-10	-3	4	11	12	16

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中
动态规划法

Needleman-Wunsch

按照动态规划算法填满表格：

	GAP	A	T	C	C	G
GAP	0	-2	-4	-6	-8	-10
A	-2	5	3	1	-1	-3
G	-4	3	4	2	0	4
T	-6	1	8	6	4	2
C	-8	-1	6	13	11	9
G	-10	-3	4	11	12	16

全局比对最优路径：右下方最后一个位置回溯到左上方位置值为0的路径。

PART TWO 4

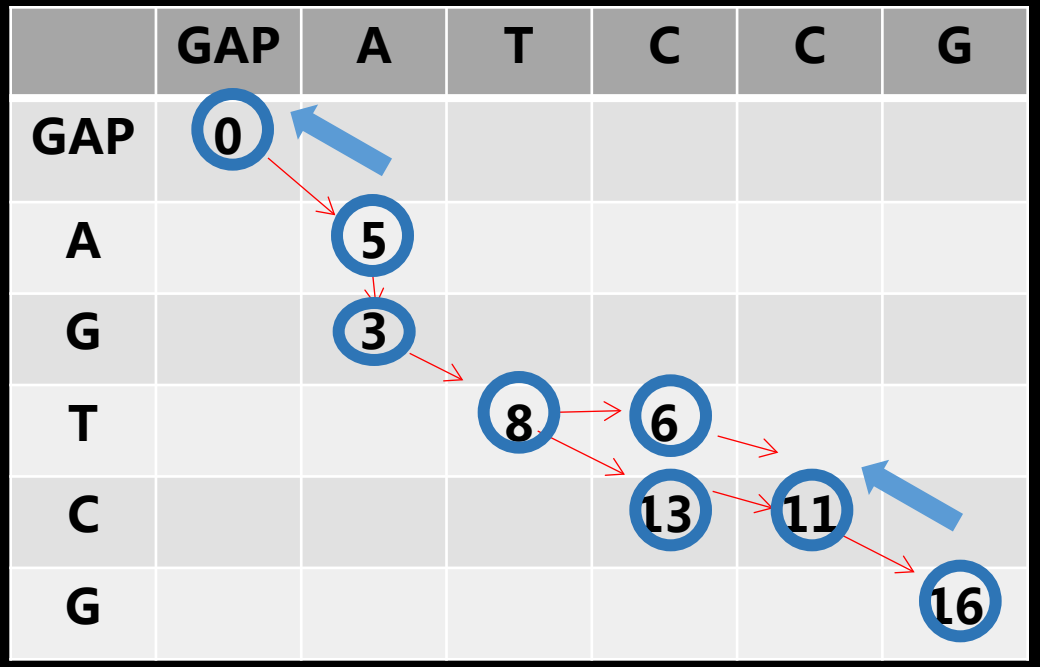
双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中

动态规划法

Needleman-Wunsch



全局比对最优路径：

右下方最后一个位

置回溯到左上方位

置值为0的路径。

体现核心思想：

最好比对 =

之前最好比对+当前最好比对

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

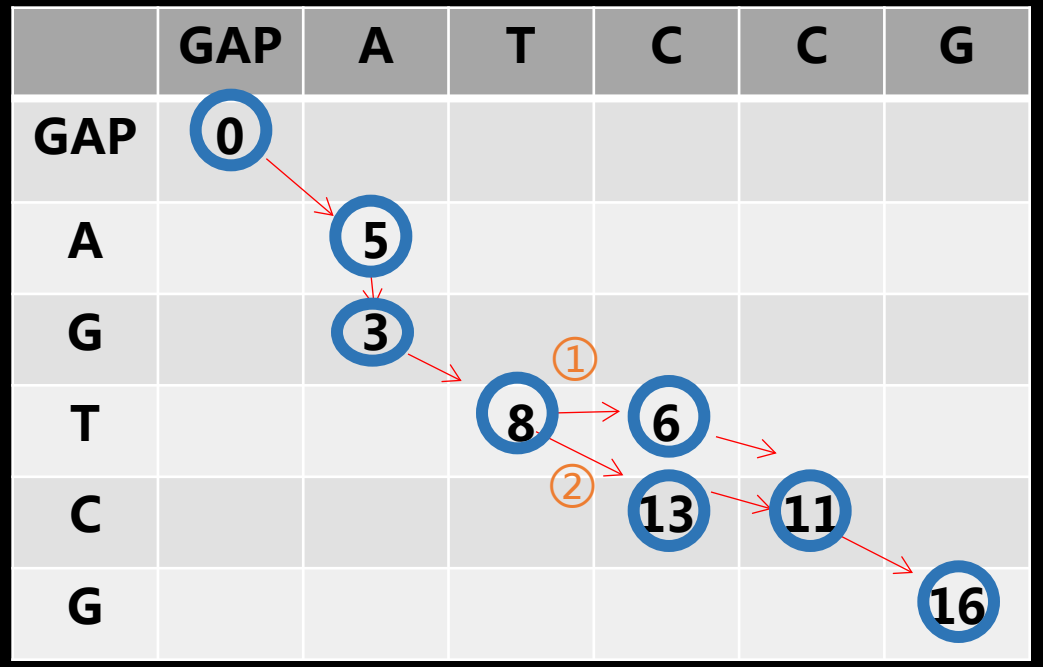
2.1

全局比对中

动态规划法

Needleman-Wunsch

两条“最优”路径：



第一条比对：

A-TCCG
AGT-CG

第二条比对：

A-TCCG
AGTC-G

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.1

全局比对中
动态规划法

Needleman-Wunsch

小结：Needleman-Wunsch算法

1.填满矩阵：计算次数n*n

2.回溯路径：矩阵值确定即可得“最优”路径

问题：长度为10的双序列比对， Needleman-Wunsch算法运算速度是穷举算法的多少倍？

答案： Needleman-Wunsch算法：10*10=100

穷举法： C_{20}^{10} **=184,756**

倍数：184，756/100=1847.56倍

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.2

局部比对中

动态规划法

Smith-Waterman

局部比对动态算法：

$$F(0,0)=0$$

$$F(i,j)=\max\begin{cases}F(i-1,j-1)+s(x_i,y_j)\\F(i-1,j)+d\\F(i,j-1)+d\\0\end{cases}$$

增加了一个选项0，
可保证*F(i,j)*不小于0

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.2

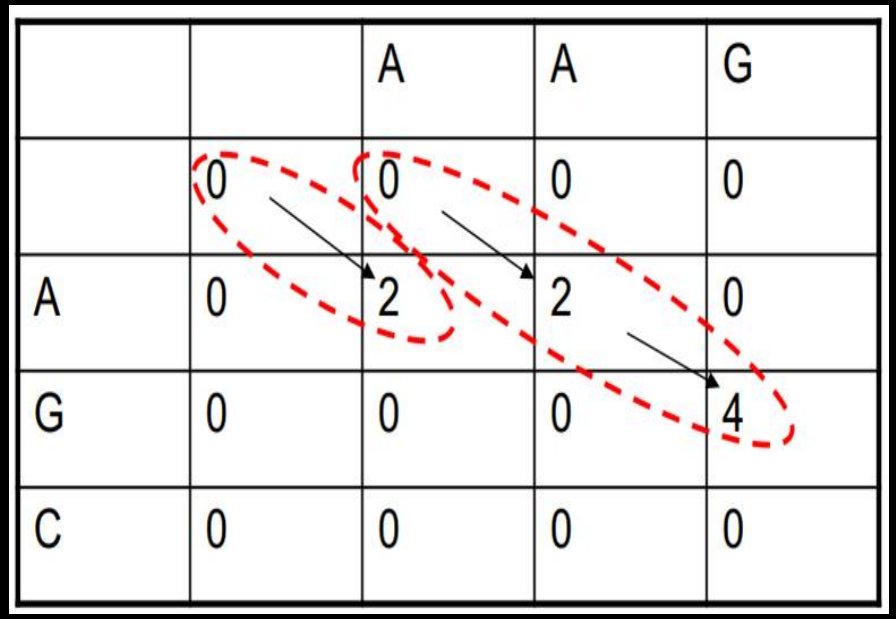
局部比对中

动态规划法

Smith-Waterman

举个例子：比对AAG与AGC

标记部分为“最优”路径



		A	A	G
	0	0	0	0
A	0	2	2	0
G	0	0	0	4
C	0	0	0	0

体现局部比对思想：

可形成和保留局部的

高分比对区域

PART TWO 4

双序列比对算法 2-D Alignment Algorithms

2.2

局部比对中

动态规划法

Smith-Waterman

举个例子：比对AAG与AGC

		A	A	G
	0	0	0	0
A	0	2	2	0
G	0	0	0	4
C	0	0	0	0

第一条比对：

A

A

第二条比对：

AG

AG

PART TWO 4

序列比对算法 Alignment Algorithms

3

需求支撑：

Query序列长度为300bp —动态规划算法→ 计算量：300*N (N为数据库的序列长度）

BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

解决方法：

将大量序列数据的一部分小片段(种子：seed)存储在超快的检索数据结构中（如hash）先利用seed进行粗定位，再从seed锚定位点处向两侧延伸比对，延伸过程可采用精细的动态规划算法。这将极大的节省计算时间。

PART TWO 4

序列比对算法 Alignment Algorithms

3

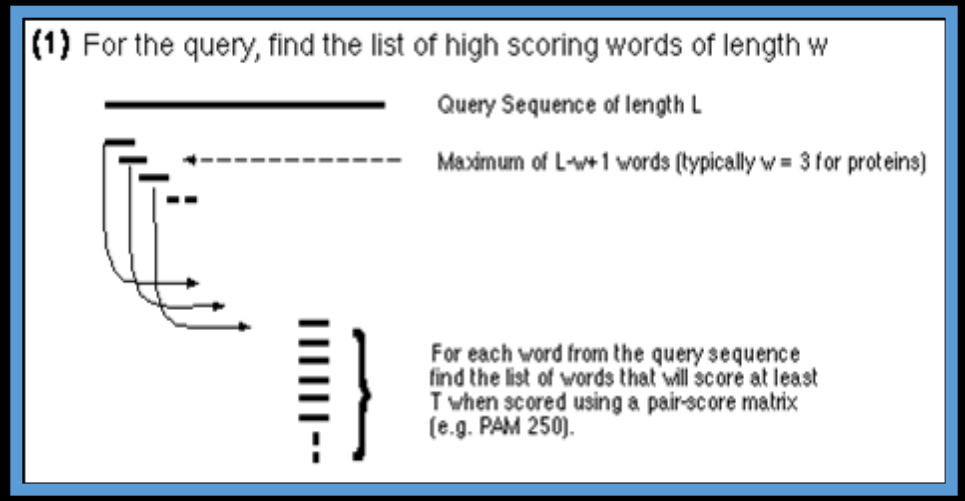
BLAST

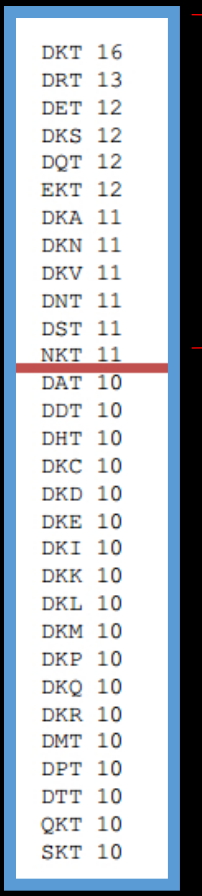
基于种子定位的比对算法

seed and extension

（启发式算法）

第一步：找出每个单词按照打分矩阵得分高于临界值的所有邻居字串





得分大于11的邻居字串

举个例子

Query长度：15，共分成13个单位；以其中

DKT单位为例按照BLOSUME62得分列表如右：

PART TWO 4

序列比对算法 Alignment Algorithms

3

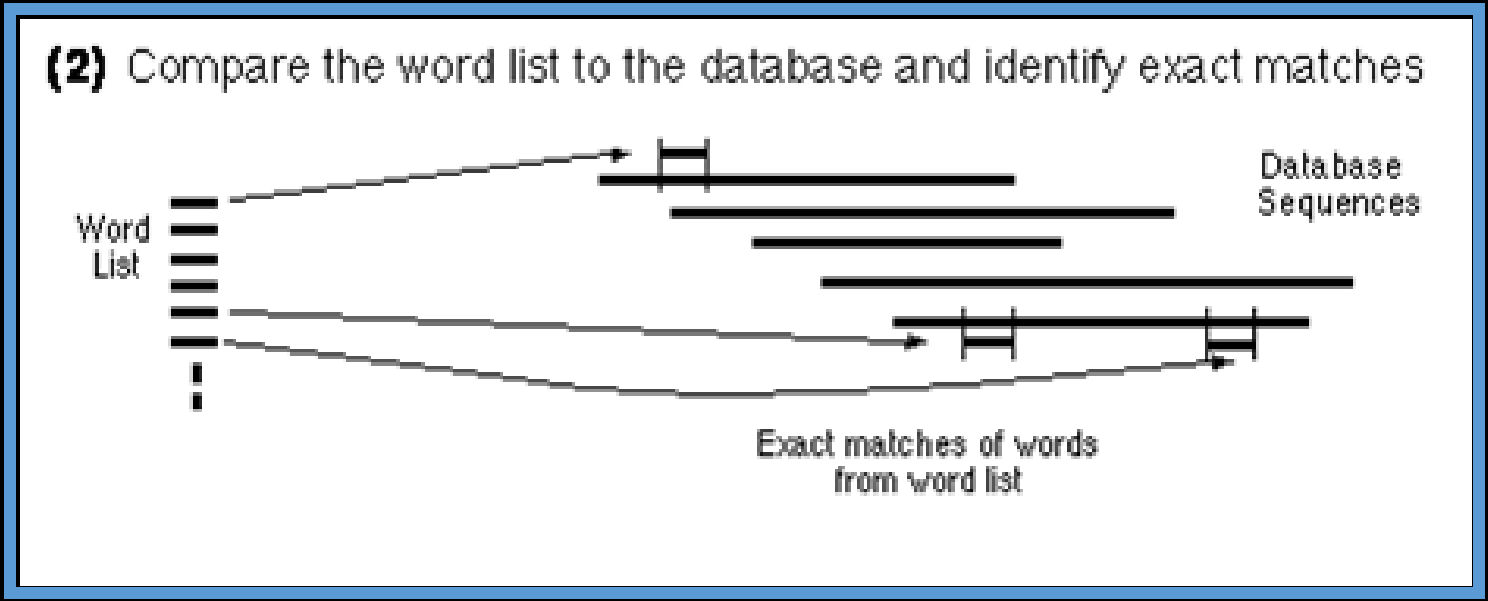
BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

第二步：将Query的单位列表与数据库列表比较，确定合适的比对



Compare word lists by Hashing（哈希算法）

PART TWO 4

序列比对算法 Alignment Algorithms

3

第三步：将合适的比对向两端延伸，得到得分大于临界值的高得分片段HSP

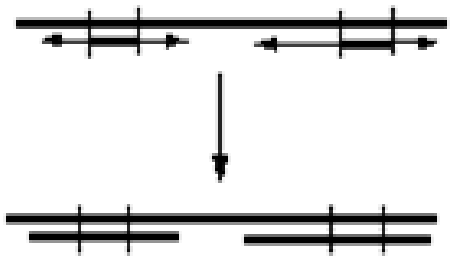
BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

(3) For each word match, extend the alignment in both directions to find alignments that score greater than a threshold of value S



Maximal Segment Pairs (MSPs)

PART TWO 4

序列比对算法 Alignment Algorithms

3

BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

- **HSP： High-scoring Segment Pairs**

  是根据筛选的与数据库匹配单位的列表在延伸后得到的片段。

- **BLAST 前: 一般须去除高度重复序列。**

- **BLAST后：一般得到很多短HSP而不是单一比对区域。**

PART TWO 4

序列比对算法 Alignment Algorithms

3

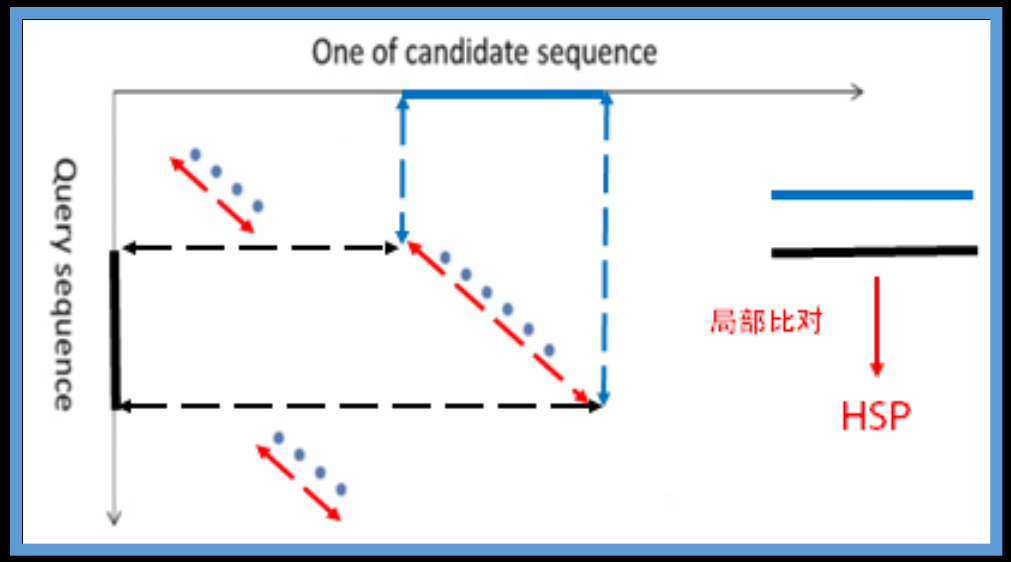
BLAST

基于种子定位的比对算法

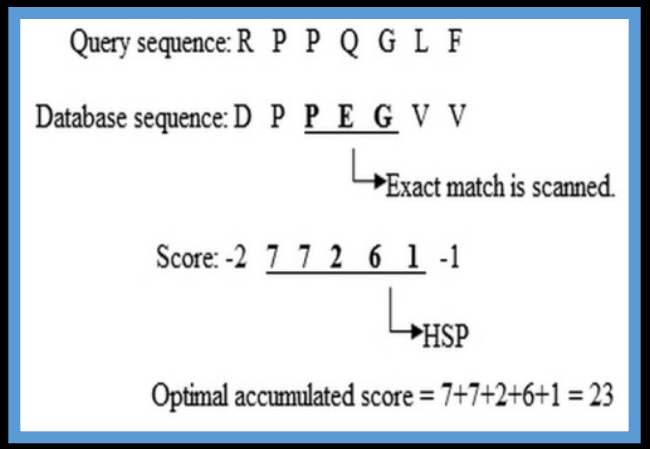
seed and extension

（启发式算法）

采用：Smith-Waterman局部比对动态规划算法



得分举例：



PART TWO 4

序列比对算法 Alignment Algorithms

3

BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

- **如何评价BLAST搜索结果： *E* values and *P* values**

计算公式：

$E = Kmn\ e^{-\lambda S}$

***E*=在数据库随机搜索发生大于或等于得分S的高得分片段**

比对数目= Expect value=错误率（*樊伟*）

m, n = the length of two sequences

λ, K =打分矩阵的标准化系数

PART TWO 4

序列比对算法 Alignment Algorithms

3

BLAST

基于种子定位的比对算法

seed and extension

（启发式算法）

●**如何评价BLAST搜索结果： *E* values and *P* values**

计算公式：

$p = 1 - e^{-E}$

两者转换关系如下：

E	*p*
10	0.99995460
5	0.99326205
2	0.86466472
1	0.63212056
0.1	0.09516258
0.05	0.04877058
0.001	0.00099950
0.0001	0.00010000

序列比对算法 Alignment Algorithms

4

PSI-BLAST

Position specific iterated

特定位置迭代BLAST

PSSM：特异位置打分矩阵

Position-Specific Scoring Matrix

Amino acid substitution scores are given separately for each position in a protein multiple sequence alignment.

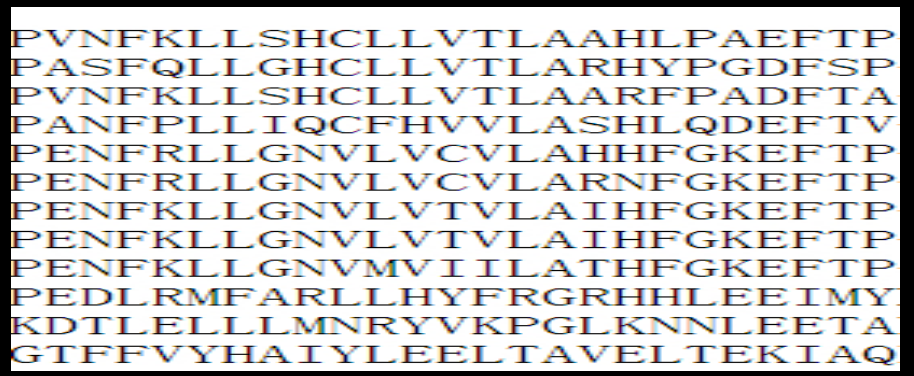
蛋白质多重序列比对中的每个位置分别给出氨基酸替代得分。

PART TWO 4

序列比对算法 Alignment Algorithms

第一步：用Blastp找出序列

第二步：比对



第三步：得到频率矩阵

Amino Acid	1	2	3	4	5	6	7
N	0.17	0.04	0.04	0.04	0.04	0.04	0.04
T	0.04	0.09	0.09	0.04	0.04	0.04	0.04
E	0.04	0.04	0.09	0.04	0.13	0.09	0.04
G	0.04	0.04	0.04	0.13	0.09	0.04	0.09
W	0.04	0.04	0.04	0.04	0.04	0.09	0.09
I	0.04	0.13	0.04	0.04	0.04	0.04	0.04
H	0.04	0.04	0.04	0.04	0.04	0.04	0.04
R	0.04	0.04	0.04	0.09	0.04	0.04	0.04
A	0.04	0.04	0.09	0.04	0.04	0.04	0.04
C	0.04	0.04	0.04	0.04	0.04	0.09	0.09
...	...	...	...	...	...	...	...

第四步：得到打分矩阵

Amino Acid	1	2	3	4	5	6	7
N	0.54	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
T	-0.06	0.24	0.24	-0.06	-0.06	-0.06	-0.06
E	-0.06	-0.06	0.24	-0.06	0.42	0.24	-0.06
G	-0.06	-0.06	-0.06	0.42	0.24	-0.06	0.24
W	-0.06	-0.06	-0.06	-0.06	-0.06	0.24	0.24
I	-0.06	0.42	-0.06	-0.06	-0.06	-0.06	-0.06
H	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
R	-0.06	-0.06	-0.06	0.24	-0.06	-0.06	-0.06
A	-0.06	-0.06	0.24	-0.06	-0.06	-0.06	-0.06
C	-0.06	-0.06	-0.06	-0.06	-0.06	0.24	0.24
...	...	...	...	...	...	...	...

PART TWO 4

序列比对算法 Alignment Algorithms

频率矩阵

NTEGEWI
NITRGEW
NIAGECC

The observed frequency of 'I' occurring at position 2 is 2/3, or .67

$$Freq = \frac{N_{obs}}{\sum N_{obs}}$$

Amino Acid	1	2	3	4	5	6	7
N	1.00	0.00	0.00	0.00	0.00	0.00	0.00
T	0.00	0.33	0.33	0.00	0.00	0.00	0.00
E	0.00	0.00	0.33	0.00	0.67	0.33	0.00
G	0.00	0.00	0.00	0.67	0.33	0.00	0.33
W	0.00	0.00	0.00	0.00	0.00	0.33	0.33
I	0.00	0.67	0.00	0.00	0.00	0.00	0.00
H	0.00	0.00	0.00	0.00	0.00	0.00	0.00
R	0.00	0.00	0.00	0.33	0.00	0.00	0.00
A	0.00	0.00	0.33	0.00	0.00	0.00	0.00
C	0.00	0.00	0.00	0.00	0.00	0.33	0.33
...	...	...	...	...	...	...	...

（高歌 降帅）

序列比对算法 Alignment Algorithms

对频率矩阵进行调整：

$$\hat{Freq} = \frac{(N_{obs} + 1)}{\sum N_{obs} + 20} = \frac{Freq \times \sum N_{obs} + 1}{\sum N_{obs} + 20}$$

PART TWO 4

序列比对算法 Alignment Algorithms

频率矩阵调整

Amino Acid	1	2	3	4	5	6	7
N	1.00	0.00	0.00	0.00	0.00	0.00	0.00
T	0.00	0.33	0.33	0.00	0.00	0.00	0.00
E	0.00	0.00	0.33	0.00	0.67	0.33	0.00
G	0.00	0.00	0.00	0.67	0.33	0.00	0.33
W	0.00	0.00	0.00	0.00	0.00	0.33	0.33
I	0.00	0.67	0.00	0.00	0.00	0.00	0.00
H	0.00	0.00	0.00	0.00	0.00	0.00	0.00
R	0.00	0.00	0.00	0.33	0.00	0.00	0.00
A	0.00	0.00	0.33	0.00	0.00	0.00	0.00
C	0.00	0.00	0.00	0.00	0.00	0.33	0.33
...	...	...	...	...	...	...	...

NTEGEWI
NITRGEW
NIAGECC

$$\frac{3\times.067+1}{3+20}\approx 0.13$$

Amino Acid	1	2	3	4	5	6	7
N	0.17	0.04	0.04	0.04	0.04	0.04	0.04
T	0.04	0.09	0.09	0.04	0.04	0.04	0.04
E	0.04	0.04	0.09	0.04	0.13	0.09	0.04
G	0.04	0.04	0.04	0.13	0.09	0.04	0.09
W	0.04	0.04	0.04	0.04	0.04	0.09	0.09
I	0.04	0.13	0.04	0.04	0.04	0.04	0.04
H	0.04	0.04	0.04	0.04	0.04	0.04	0.04
R	0.04	0.04	0.04	0.09	0.04	0.04	0.04
A	0.04	0.04	0.09	0.04	0.04	0.04	0.04
C	0.04	0.04	0.04	0.04	0.04	0.09	0.09
...	...	...	...	...	...	...	...

（高歌 降帅）

PART TWO 4

序列比对算法 Alignment Algorithms

打分矩阵: PSSM

$$odds\ ratio = \frac{P(x \mid M)}{P(x \mid R)}$$

$P(x \mid M)$ ——> 调整后的频率矩阵的值

$P(x \mid R)$ ——> 在此位置随机出现概率

$(1/20)^n$ n: 序列长度

$$Score = \log odds\ ratio = \log\left(\frac{P(x \mid M)}{P(x \mid R)}\right) = \log(P(x \mid M) - \log(P(x \mid R))$$

（高歌 降帅）

PART TWO 4

序列比对算法 Alignment Algorithms

PSSM

Amino Acid	1	2	3	4	5	6	7
N	0.17	0.04	0.04	0.04	0.04	0.04	0.04
T	0.04	0.09	0.09	0.04	0.04	0.04	0.04
E	0.04	0.04	0.09	0.04	0.13	0.09	0.04
G	0.04	0.04	0.04	0.13	0.09	0.04	0.09
W	0.04	0.04	0.04	0.04	0.04	0.09	0.09
I	0.04	0.13	0.04	0.04	0.04	0.04	0.04
H	0.04	0.04	0.04	0.04	0.04	0.04	0.04
R	0.04	0.04	0.04	0.09	0.04	0.04	0.04
A	0.04	0.04	0.09	0.04	0.04	0.04	0.04
C	0.04	0.04	0.04	0.04	0.04	0.09	0.09
...	...	...	...	...	...	...	...

NTEGEWI
NITRGEW
NIAGECC

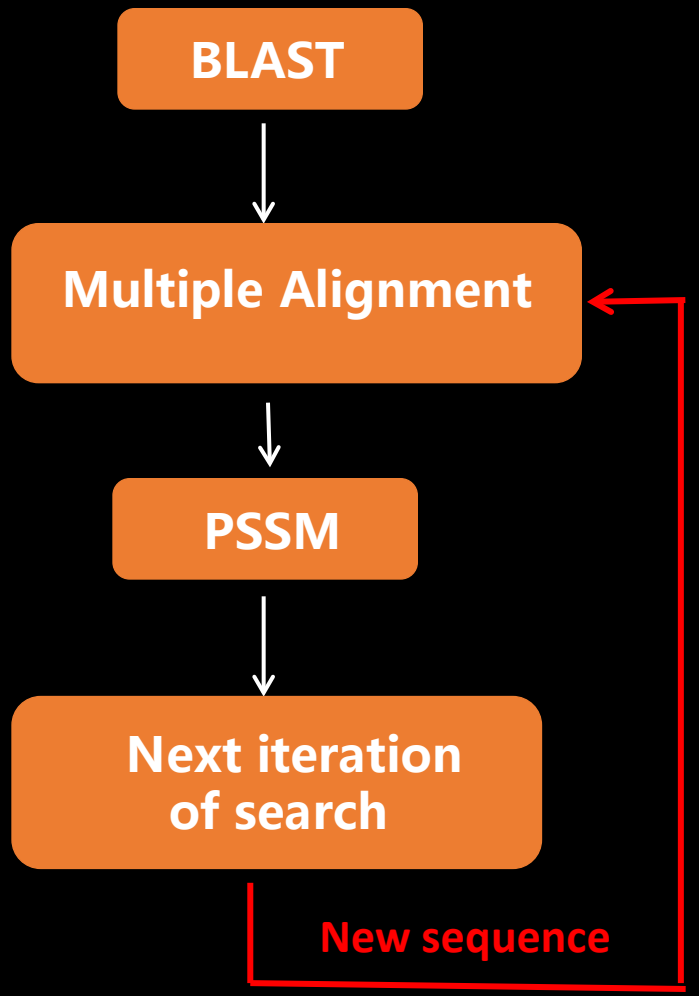
$$\log\left(\frac{0.13}{0.05}\right) \approx 0.42$$

Amino Acid	1	2	3	4	5	6	7
N	0.54	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
T	-0.06	0.24	0.24	-0.06	-0.06	-0.06	-0.06
E	-0.06	-0.06	0.24	-0.06	0.42	0.24	-0.06
G	-0.06	-0.06	-0.06	0.42	0.24	-0.06	0.24
W	-0.06	-0.06	-0.06	-0.06	-0.06	0.24	0.24
I	-0.06	0.42	-0.06	-0.06	-0.06	-0.06	-0.06
H	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06	-0.06
R	-0.06	-0.06	-0.06	0.24	-0.06	-0.06	-0.06
A	-0.06	-0.06	0.24	-0.06	-0.06	-0.06	-0.06
C	-0.06	-0.06	-0.06	-0.06	-0.06	0.24	0.24
...	...	...	...	...	...	...	...

（高歌 降帅）

PART TWO 4

序列比对算法 Alignment Algorithms



（降帅）

PART TWO 4

序列比对算法 Alignment Algorithms

5

思考

① 什么是多序列比对算法？

② 什么是Hidden Markov models？

（隐马尔可夫模型）

③ 什么是基于profile的比对算法？

PART THREE

应用讨论

PART Three 1

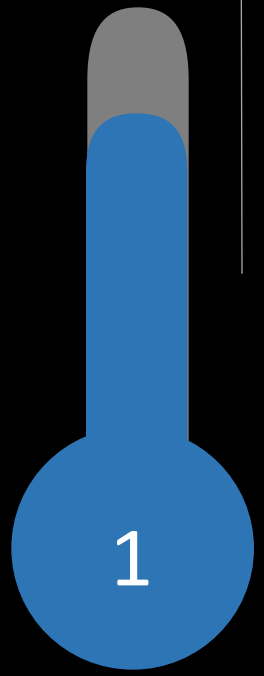
为什么使用BLAST？

BLAST searching is fundamental to understanding the relatedness of any favorite query sequence to other known proteins or DNA sequences.

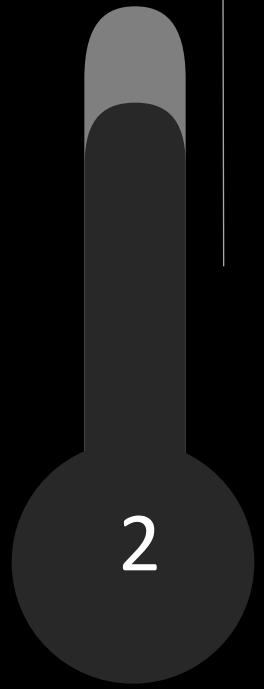
BLAST是理解搜寻序列与其他蛋白或DNA序列相关性的快速算法

PART Three 2

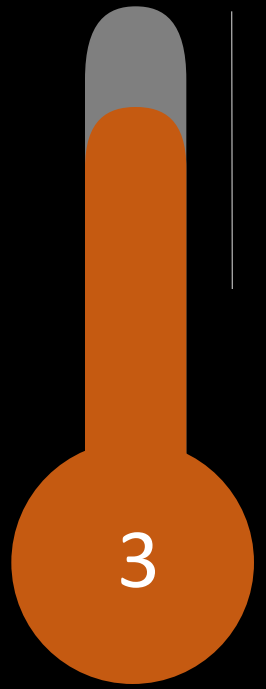
应用 Applications



确定直系同源和旁系同源



发现新基因、新蛋白或者发现基因、蛋白的变异体



研究表达序列标签（EST)；探索蛋白质结构和功能

PART Three 3

BLAST程序家族 Programs family

Query	Database	Program	Comment
Protein	Protein	blastp fastp	
Nucleotide	Nucleotide	blastn fastn	Use only if nucleotide comparison is really wanted
Nucleotide	Protein	blastx fastx3	Translate query to protein; 6-frame
Protein	Nucleotide	tblastn tfastx3	Translate DB on the fly; 6-frame
Nucleotide	Nucleotide	tblastx	Translate both query and DB (gene-oriented); 2* 6-frame

http://biomedicum.ut.ee/~kraulis

PART

参考资料 Reference

- **约翰霍普金斯大学医学院Jonathan Pevsner, Ph.D.关于BLAST介绍**
  http://www.bioinfbook.org

- **ABC网站上北京大学生命科学院高歌、**亢雨笺、降帅关于BLAST的介绍
  http://abc.cbi.pku.edu.cn/

- **深圳农业基因组研究所樊伟关于序列比对课件**

THANKS TO...

BLAST算法讨论小组全体成员：

董喆　　**王岩岩**

王恒超　　**黄雾**

唐蝶　　**朱安东**

指导老师：　罗静初

扫一扫，加好友

@实用生物信息技术专题讨论2016~

Applied Bioinformatics Course Seminar