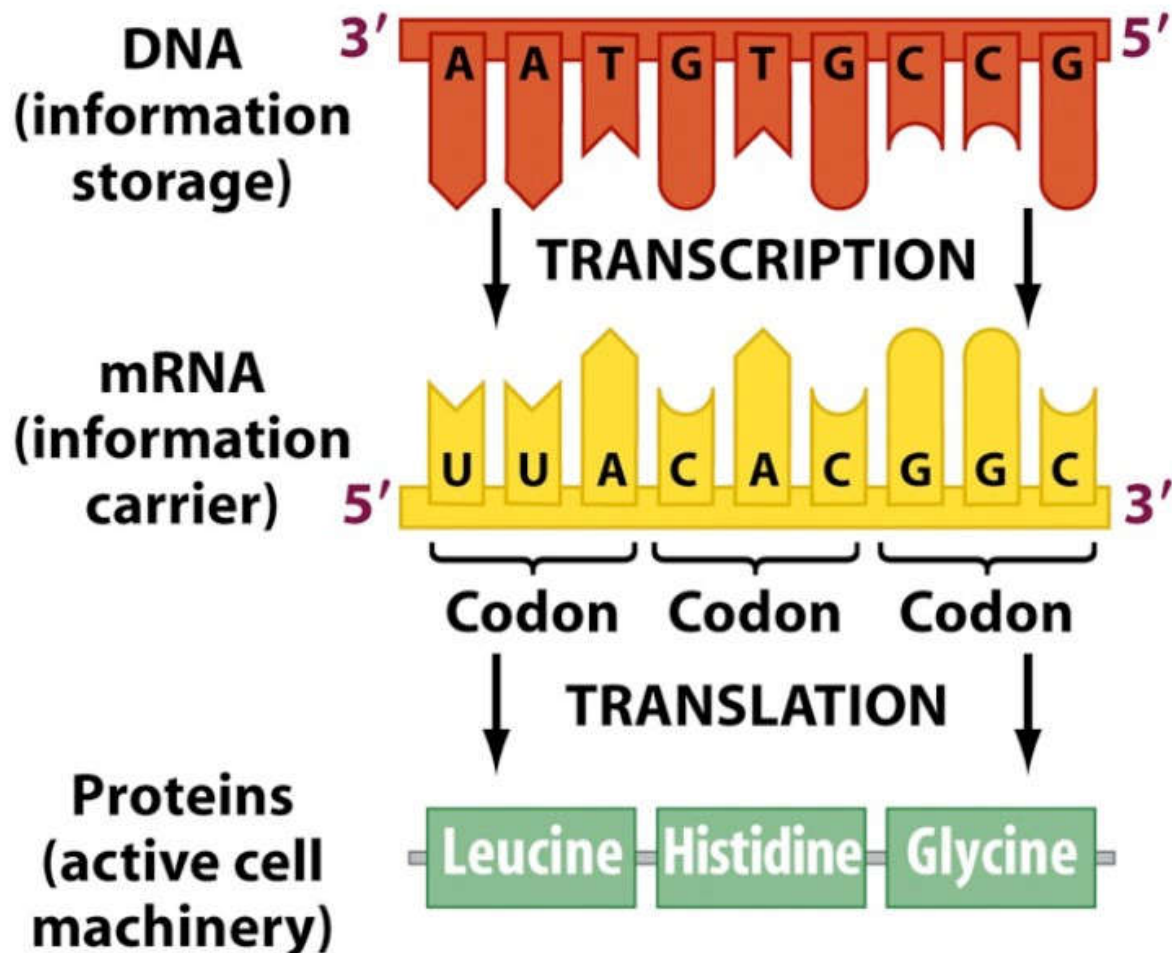


Выравнивания последовательностей

Центральная Догма

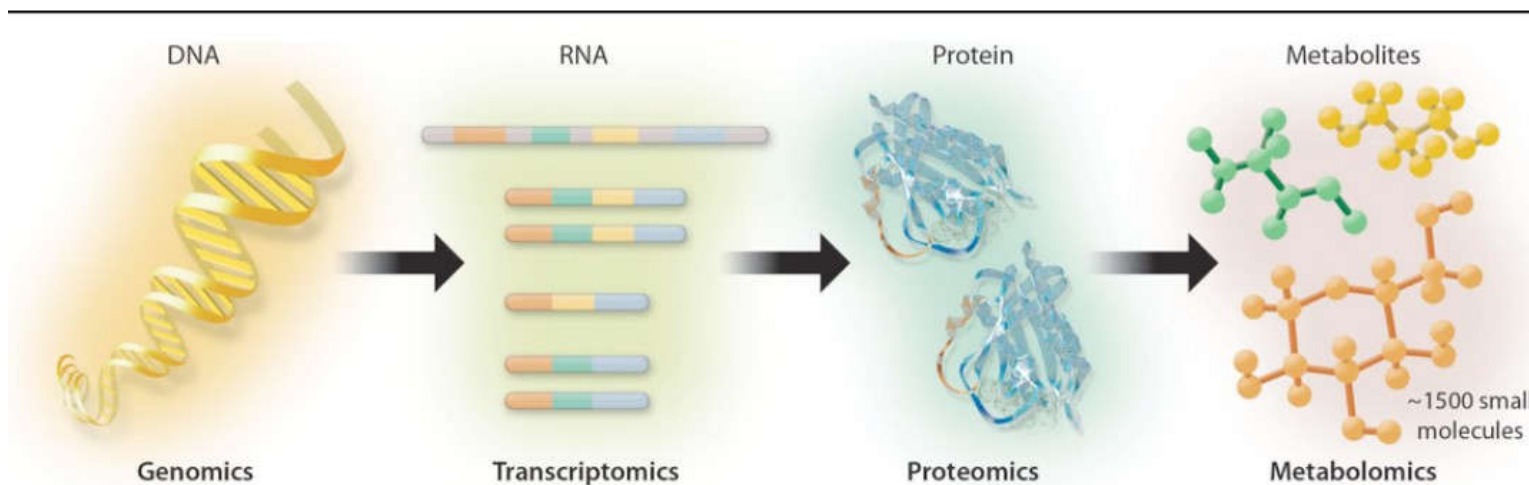


Омики

Геном — все ДНК организма.

Транскриптом — все РНК организма.

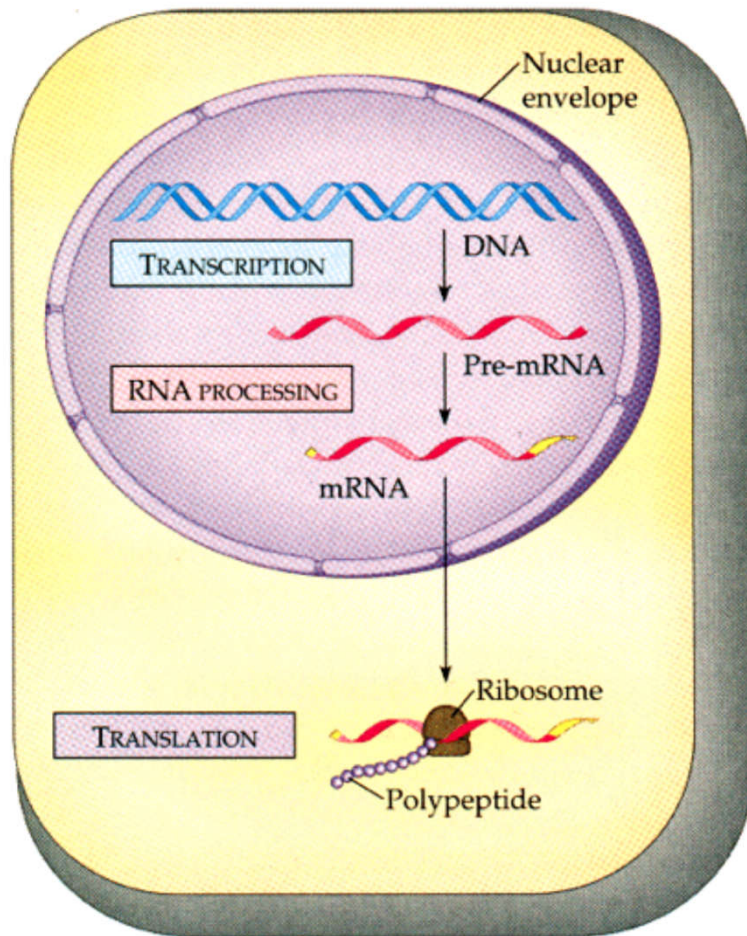
Протеом — все белки организма.



Выравнивание последовательностей (строк)

- Точечные графики
- Редакционное расстояние
- Матрицы замен
- Глобальное и локальное выравнивание
- Аффинная модель вставки
- Линейная память и 4 русских
- Множественное выравнивание
- FASTA, BLAST

Гены



CCTGAGCCAAC TATTGATGAA

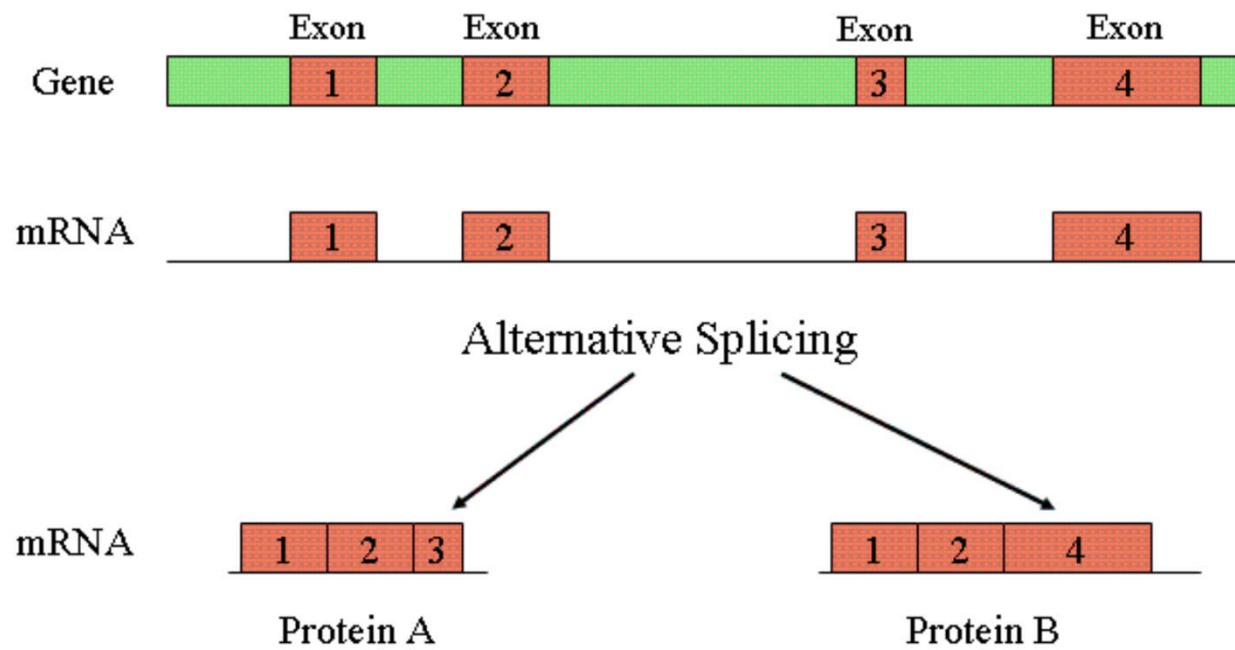


CCUGAGCCAACUAUUGAUGAA



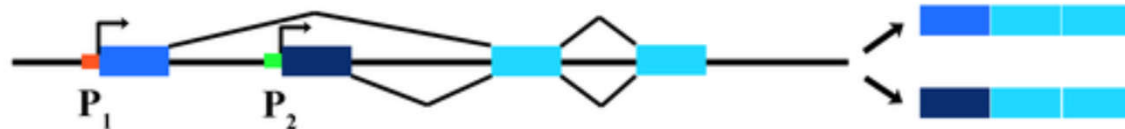
PEPTIDE

Сплайсинг



Сплайсинг

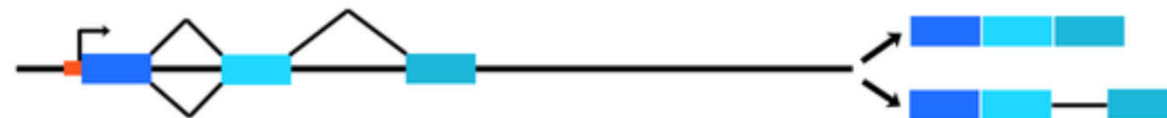
(a) Alternative selection of promoters (e.g., *myosin* primary transcript)



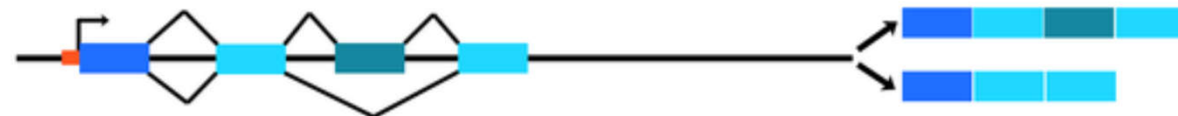
(b) Alternative selection of cleavage/polyadenylation sites (e.g., *tropomyosin* transcript)



(c) Intron retaining mode (e.g., *transposase* primary transcript)

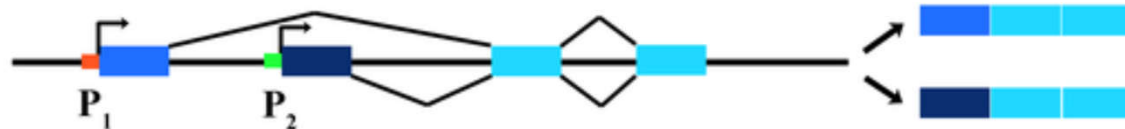


(d) Exon cassette mode (e.g., *troponin* primary transcript)



Сплайсинг

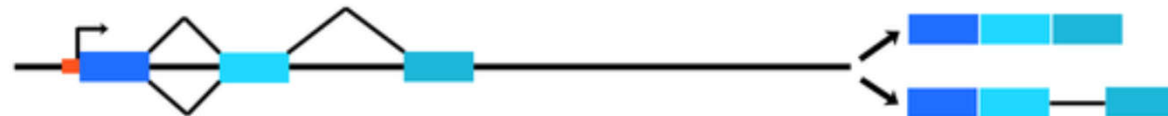
(a) Alternative selection of promoters (e.g., *myosin* primary transcript)



(b) Alternative selection of cleavage/polyadenylation sites (e.g., *tropomyosin* transcript)



(c) Intron retaining mode (e.g., *transposase* primary transcript)

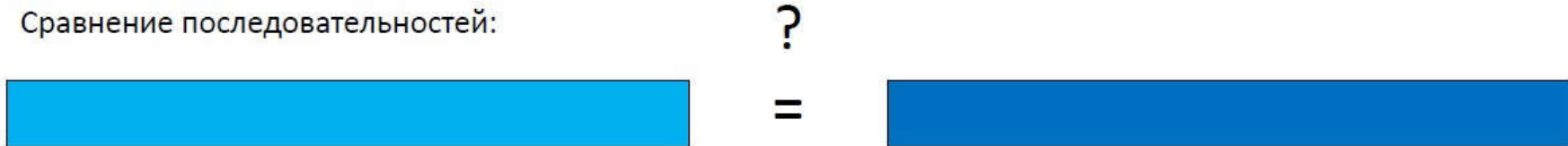


(d) Exon cassette mode (e.g., *troponin* primary transcript)



Выравнивание последовательностей (строк)

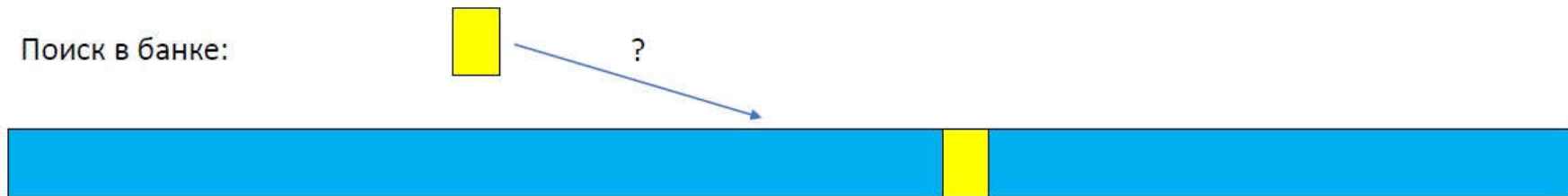
Сравнение последовательностей:



Сравнение доменов:



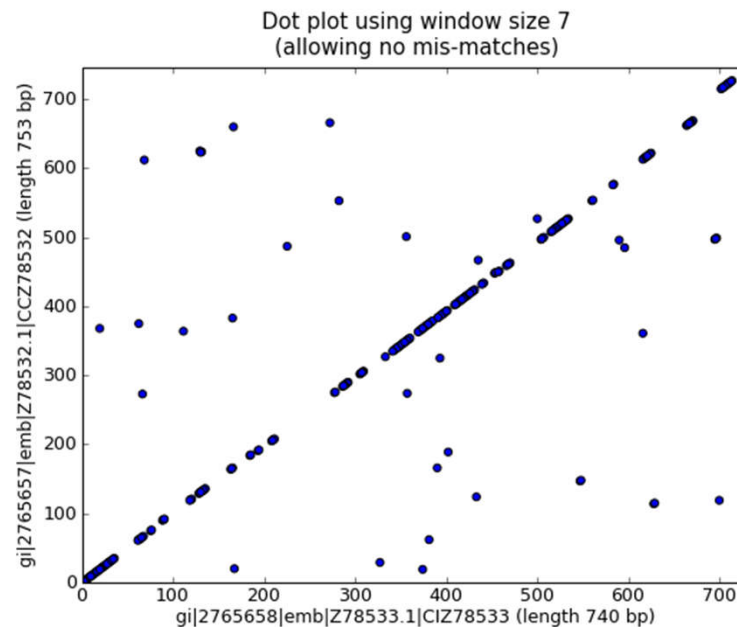
Поиск в банке:



Точечные графики

Точечные графики – это двумерные графики, показывающие схожесть двух последовательностей.

Оси графика представляют сравниваемые последовательности. Каждый регион одной последовательности сравнивается с регионом другой. <https://myhits.isb-sib.ch/cgi-bin/dotlet>

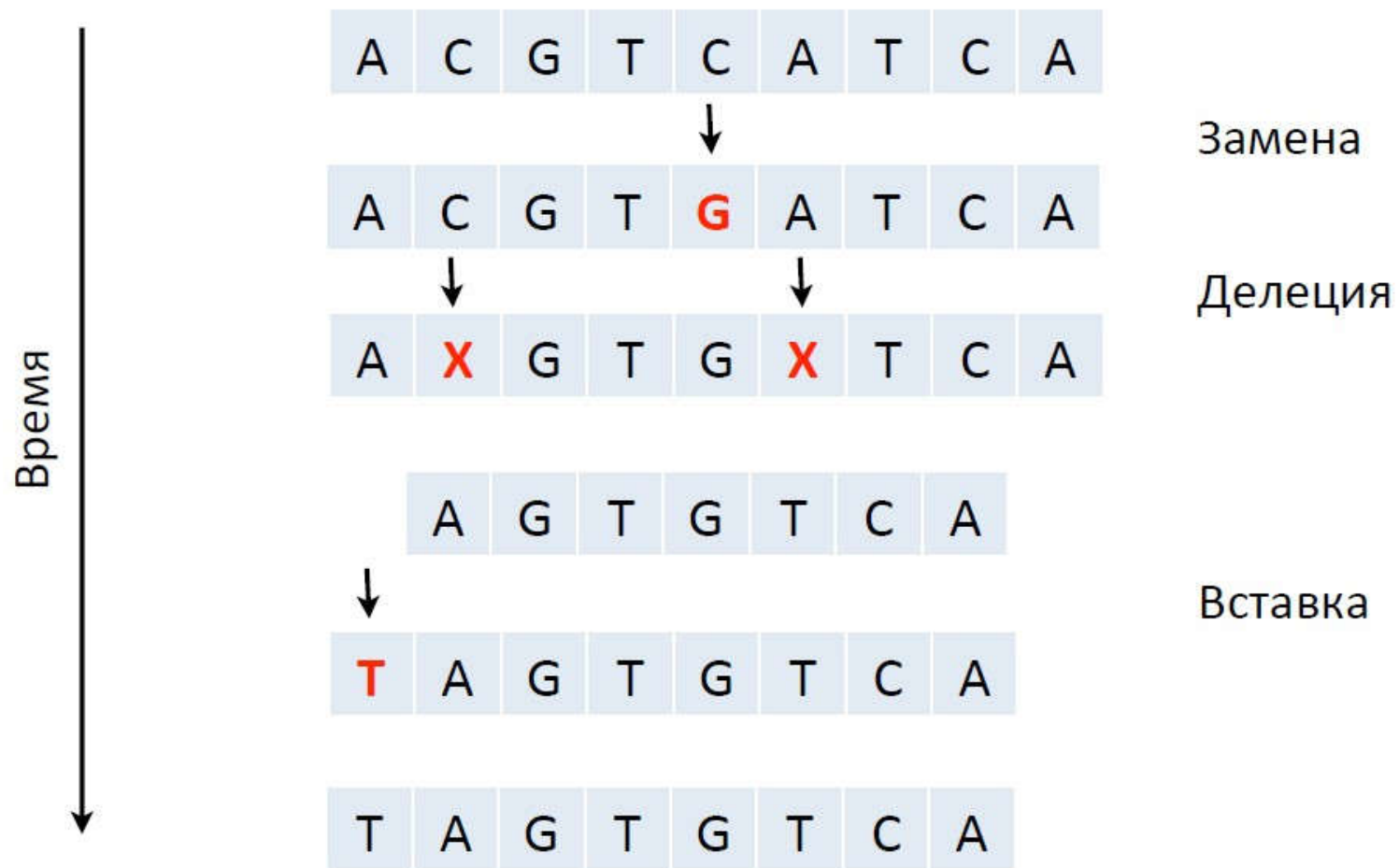


Расстояние Хэмминга

Число замен, необходимых для преобразования первой строки во вторую.

CCAGGAC
CCAAGGCT

Изменение геномов с течением времени



Редакционное расстояние (расстояние Левенштейна)

Элементарное преобразование последовательности:

замена буквы или удаление буквы или вставка буквы.

Редакционное расстояние: минимальное количество элементарных преобразований, переводящих одну последовательность в другую.

CCAGAGAC–
CCA–AGGCT

выравнивание (alignment)

редакционное предписание

МОЖЕТ ИМЕТЬ МАКСИМАЛЬНО $n + m$ КОЛОНКИ

Выравнивание последовательностей (строк)

A	C	G	T	C	A	T	C	A
---	---	---	---	---	---	---	---	---

?



Формальная постановка задачи:
определение редакционного
расстояния - минимального
количества элементарных
преобразований (замен, вставок,
делеций), переводящих одну
последовательность в другую

T	A	G	T	G	T	C	A
---	---	---	---	---	---	---	---

Выравнивание последовательностей (строк)

Рассмотрим три выравнивания двух строк

AATCTATA
AAG-AT-A

AATCTATA
AA-G-ATA

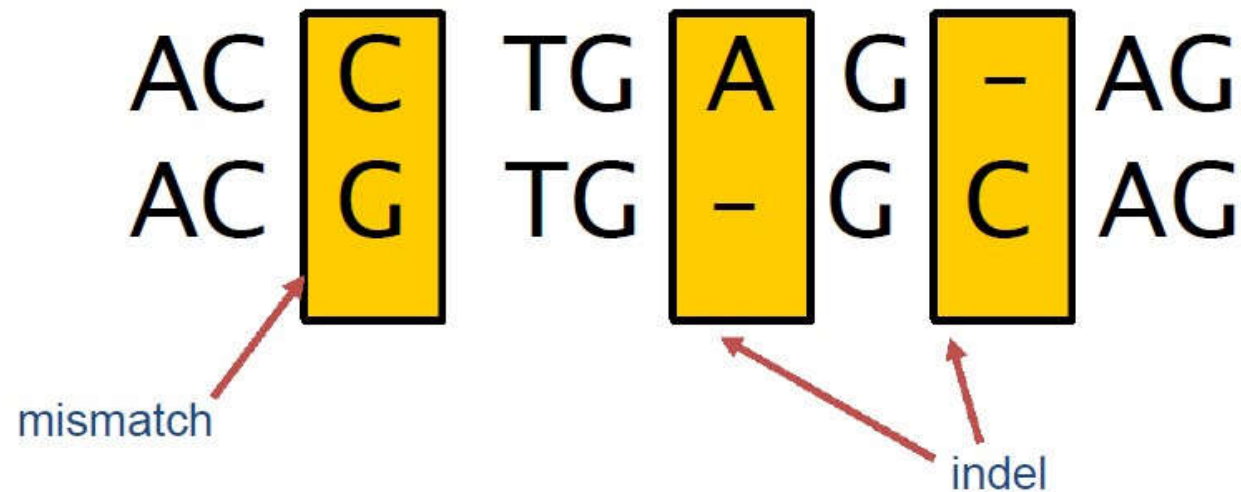
AATCTATA
AA--GATA

Редакционное расстояние рассчитывается как:

За совпадение букв	0
За несовпадение	+1
За вставку/удаление	+1

Выравнивание последовательностей (строк)

Identity



Identity = 70%

Выравнивание последовательностей (строк)

Что необходимо для вычисления оптимального выравнивания?

S1	A	C	G	T	C	A	T	C	A
S2	T	A	G	T	G	T	C	A	

- Весовая функция (scoring function)
 - Вес выравнивания = стоимость редактирования S1 в S2
 - Стоимость замены, вставки, делеции
 - Бонус за совпадение букв
- Алгоритм нахождения оптимального выравнивания
 - Перебор?

Простое взвешивание выравнивания

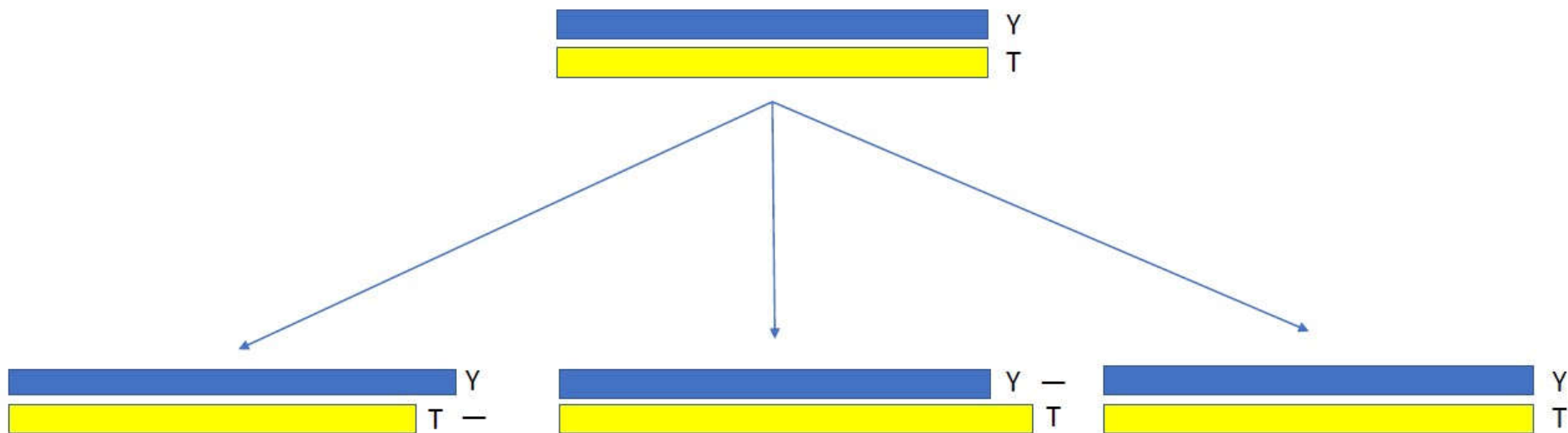
- $+1$: вес совпадения
- $-\mu$: штраф за несовпадение
- $-\delta$: штраф за делецию/вставку

Вес выравнивания = $\# \text{совпадений} - \mu(\# \text{несовпадений}) - \delta(\# \text{делеций/вставок})$

Задача – найти выравнивание с максимальным весом.

Выравнивание последовательностей (строк)

Динамическое программирование — способ решения сложных задач путём разбиения их на более простые подзадачи.



Матричное представление выравнивания

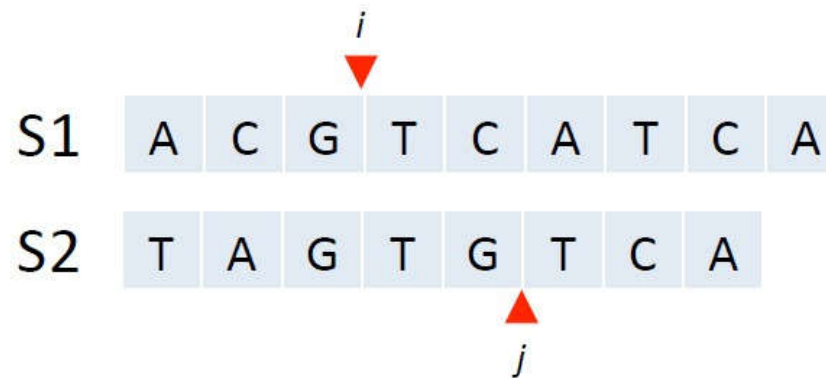
-	A	C	G	T	C	A	T	C	A
▲	■	▼	■	■	×	▼	■	■	■
T	A	-	G	T	G	-	T	C	A

		A	C	G	T	C	A	T	C	A
	begin									
T	▲									
A		■	▼							
G				■						
T					■					
G						×	▼			
T								■		
C									■	
A										■
end										

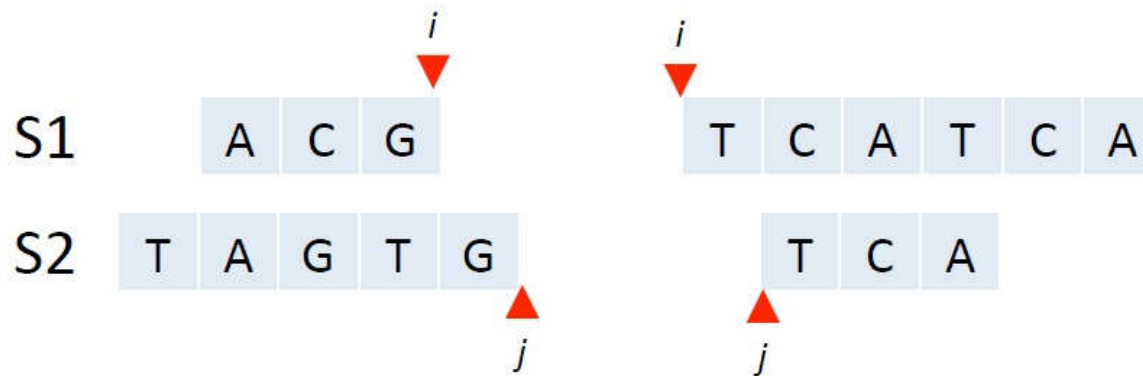
Цель: найти оптимальный путь по матрице от точки начала до точки окончания.

Матричное представление выравнивания

Вес (score) выравнивания аддитивен



Для заданного разделения (i,j) вес оптимального выравнивания есть:
вес оптимального выравнивания между $S1[1,i]$ и $S2[1,j]$ +
вес оптимального выравнивания между $S1[i,n]$ и $S2[j,m]$

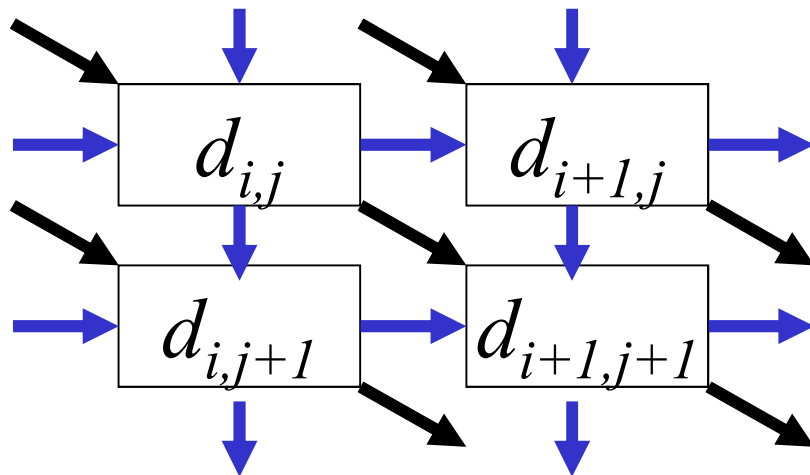


Алгоритм Нидлмана-Вунша

Граф редакционного расстояния для последовательностей S^1, S^2 :

вершина $v_{i,j}$ соответствует префиксам последовательностей $\{S^1_{1..i}\}, \{S^2_{1..j}\}$. На вершине записано редакционное расстояние между префиксами.

(синие стрелки соответствуют вставкам и удалениям)



$$d_{i+1,j+1} = \min \{ d_{i+1,j} + 1, \\ d_{i,j+1} + 1, \\ d_{i,j} + e_{i+1,j+1} \}$$

$$e_{i,j} = \{ \begin{array}{l} 0, S^1_i = S^2_j; \\ 1, S^1_i \neq S^2_j \end{array} \}$$

Алгоритм Нидлмана-Вунша

$S1$ длины N , $S2$ длины M

$D(X, Y)$ — редакционное расстояние для
подстрок $S1[1..X]$ и $S2[1..Y]$

$D(X, Y)$ — редакционное расстояние для
подстрок $S1[1..X]$ и $S2[1..Y]$

$$D(0, 0) = 0$$

$$D(i, 0) = i$$

$$D(0, j) = j$$

		C	C	A	G	A	G	A	C
	0	1	2	3	4	5	6	7	8
C	1								
C	2								
A	3								
A	4								
G	5								
G	6								
C	7								
T	8								

Алгоритм Нидлмана-Вунша

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + (S[i] \neq S[j]) \end{cases}$$

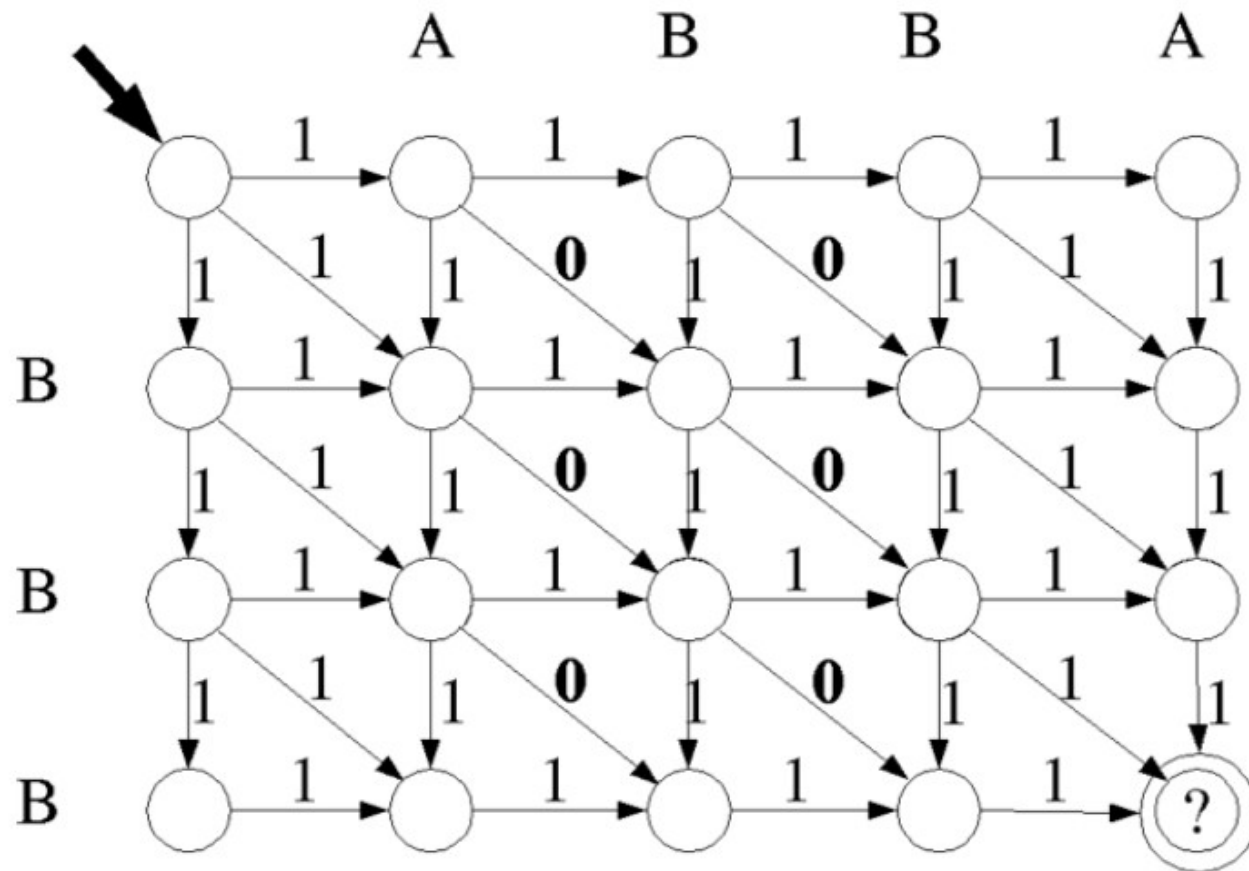
		C	C	A	G	A	G	A	C
	0	1	2	3	4	5	6	7	8
C	1	0	1	2	3	4	5	6	7
C	2	1	0	1	2	3	4	5	6
A	3	2	1	0	1	2	3	4	5
A	4	3	2	1	1	1	2	3	4
G	5	4	3	2	1	2	1	2	3
G	6	5	4	3	2	2	2	2	3
C	7	6	5	4	3	3	3	3	2
T	8	7	6	5	4	4	4	4	3

Алгоритм Нидлмана-Вунша

CCAGAGAC–
CCA–AGGCT

		C	C	A	G	A	G	A	C
	0	1	2	3	4	5	6	7	8
C	1	0	1	2	3	4	5	6	7
C	2	1	0	1	2	3	4	5	6
A	3	2	1	0	1	2	3	4	5
A	4	3	2	1	1	1	2	3	4
G	5	4	3	2	1	2	1	2	3
G	6	5	4	3	2	2	2	2	3
C	7	6	5	4	3	3	3	3	2
T	8	7	6	5	4	4	4	4	3

Путь в графе



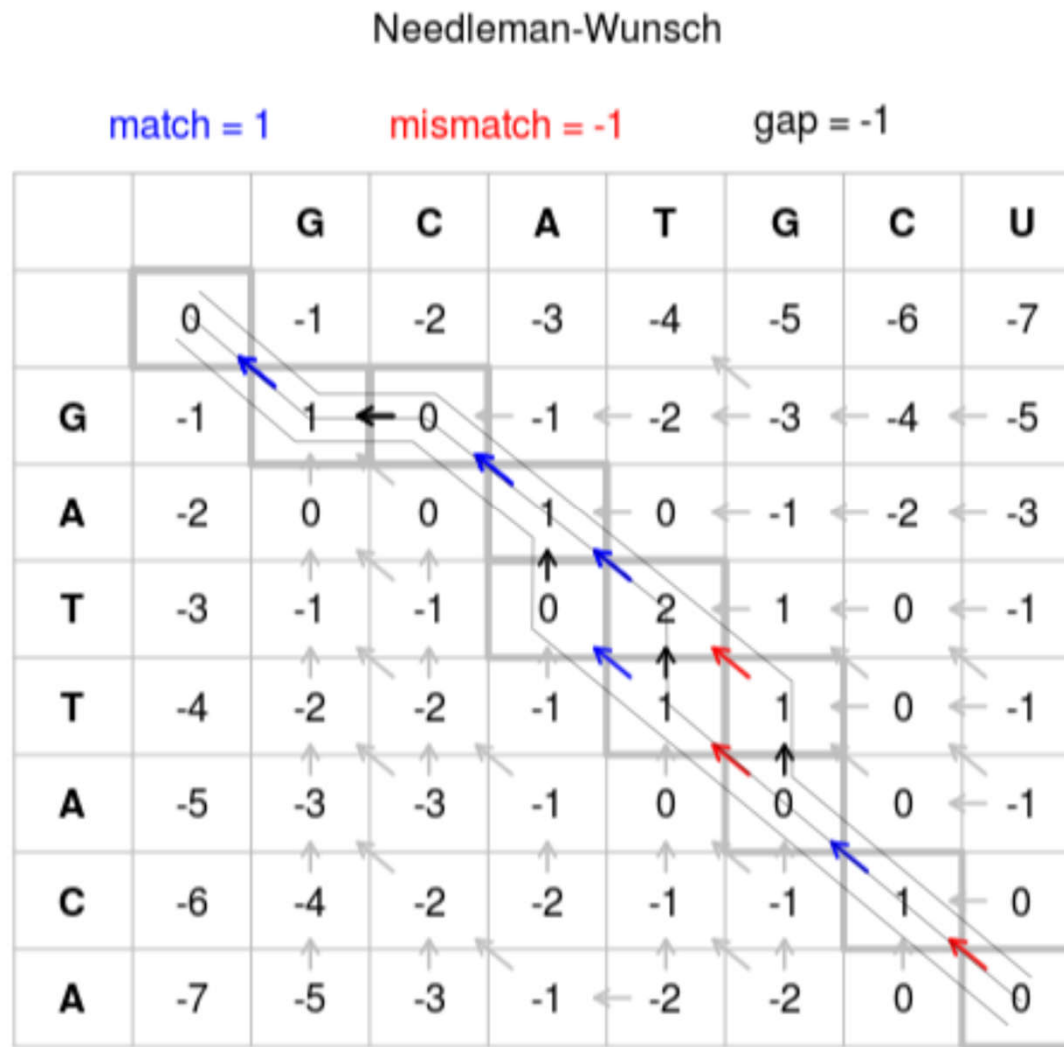
Алгоритм Нидлмана-Вунша

$$\downarrow \rightarrow = -\delta$$

$$\triangleright = \begin{cases} 1, & \text{если совпадение} \\ -\mu, & \text{если несовпадение} \end{cases}$$

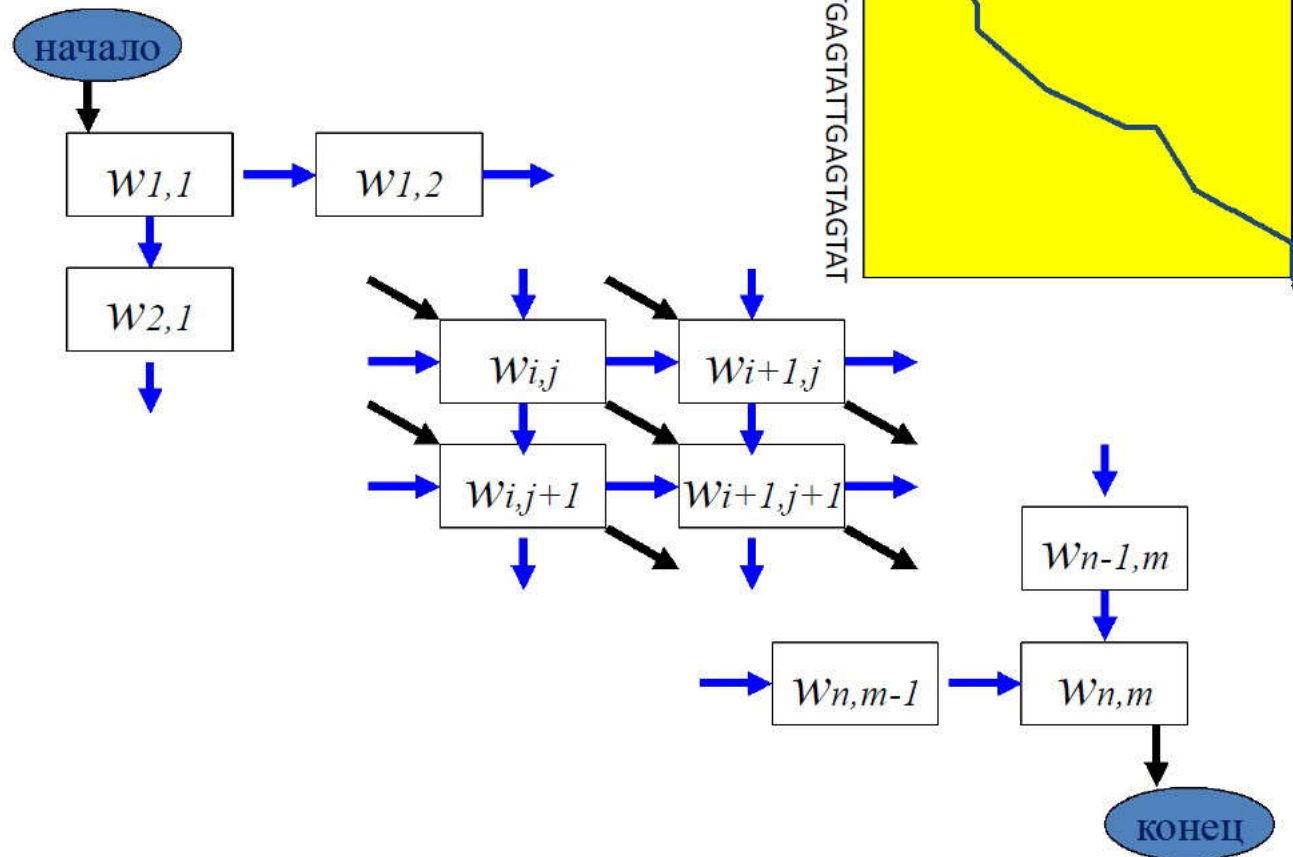
$$s_{i,j} = \max \begin{cases} s_{i-1,j-1} + 1, & \text{если } v_i = w_j \\ s_{i-1,j-1} - \mu, & \text{если } v_i \neq w_j \\ s_{i,j-1} - \delta \\ s_{i-1,j} - \delta \end{cases}$$

Алгоритм Нидлмана-Вунша



Алгоритм Нидлмана-Вунша

Алгоритм Нидлмана-Вунша



При таких граничных условиях начальные и концевые делеции штрафуются

Алгоритм Нидлмана-Вунша

При выравнивании строк разных размеров, инделы штрафуются одинаково независимо от их расположения, в некоторых случаях наиболее биологически обоснованным является наличие терминальных инделов на концах строк, как например

AACACGTGTCT
---ACGT----

В граф добавляются ребра веса 0, ведущие из начала во все граничные вершины ($i=1 \mid j=1$) и из граничных вершин ($i=n \mid j=m$) в конец



Алгоритм Нидлмана-Вунша

- Временная сложность (количество операций) -- $O(N \times M)$
- Пространственная сложность (объем памяти) -- $O(N \times M)$

Алгоритм посматривает все вершины графа

В каждой вершине делается 3 сравнения

Количество необходимых операций (время работы алгоритма): $T=O(N * M)$. Говорят, что алгоритм выравнивания квадратичен по времени работы.

Для запоминания весов и восстановления оптимального выравнивания надо в каждой вершине запомнить ее вес и направление перехода. Таким образом, алгоритм квадратичен по памяти.

Матрицы замен

Для ДНК составим $(4+1) \times (4+1)$ матрицу весов δ .

Для белков размер матрицы $(20^*+1) \times (20^*+1)$.

Дополнительные строка и столбец нужны для включения гар символа.

Это “упростит” алгоритм следующим образом:

$$s_{i,j} = \max \begin{cases} s_{i-1,j-1} + \delta(v_i, w_j) \\ s_{i-1,j} + \delta(v_i, -) \\ s_{i,j-1} + \delta(-, w_j) \end{cases}$$

Матрицы замен

- Матрицы создаются на основе экспериментальных данных.
- Выравнивания – представления белков, различающихся мутациями.
- Некоторые из этих мутаций менее пагубно влияют на функцию белка, и, соответственно, штраф $\delta(v_i, w_j)$, будет меньше прочих.

Матрицы замен

	A	R	N	K
A	5	-2	-1	-1
R	-	7	-1	3
N	-	-	7	0
K	-	-	-	6

AKRANR

KAAANK

$$-1 \ -1 \ -2 \ +5 \ +7 \ +3 = 11$$

Матрицы замен

Консервативность

Замены аминокислот, сохраняющие физико-химические свойства белков.

—Полярные на полярные

- аспартати глутамат

—Неполярные на неполярные

- аланини валин

—Прочие похожие

- лейцин и изолейцин

Типы матриц замен (весов)

- Матрицы замен аминокислот

- PAM

- BLOSUM

- ДНК матрицы

ДНК менее консервативны чем белковые последовательности, поэтому менее эффективно сравнивать кодирующие области на уровне нуклеотидов
Матрицы весов менее важны.

Типы матриц замен (весов) - PAM

- **Point Accepted Mutation** (Dayhoff et al.)
- $1\text{PAM} = \text{PAM}1 = 1\%$ аминокислот мутировали.
—Однако, после 100PAMов эволюции, не все остатки изменятся
- Некоторые остатки мутируют несколько раз
- Некоторые остатки вернутся к начальному состоянию
- Некоторые вообще не изменятся

Изначально биологи сравнивали близкие виды у которых приблизительно в среднем 1 мутация на 100 аминокислот.

Типы матриц замен (весов) - PAM

- $PAM_x = PAM_1^x$
- $PAM_{250} = PAM_1^{250}$
- PAM_{250} широко используемая матрица:

		Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	...
		A	R	N	D	C	Q	E	G	H	I	L	K	...
Ala	A	13	6	9	9	5	8	9	12	6	8	6	7	...
Arg	R	3	17	4	3	2	5	3	2	6	3	2	9	
Asn	N	4	4	6	7	2	5	6	4	6	3	2	5	
Asp	D	5	4	8	11	1	7	10	5	6	3	2	5	
Cys	C	2	1	1	1	52	1	1	2	2	2	1	1	
Gln	Q	3	5	5	6	1	10	7	3	7	2	3	5	
...														
Trp	W	0	2	0	0	0	0	0	0	1	0	1	0	
Tyr	Y	1	1	2	1	3	1	1	1	3	2	2	1	
Val	V	7	4	4	4	4	4	4	4	5	4	15	10	

Типы матриц замен (весов) - BLOSUM

- BlocksSubstitutionMatrix

- Веса извлекаются из статистики выравниваний родственных белков

- BLOSUM62 была создана на выборке последовательностей с min 62% сходством

Веса получены на основе наблюдений частоты замен в блоках локальных выравниваний соответствующих белков.

Типы матриц замен (весов) - BLOSUM

BLOSUM 62

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W

Откуда берутся параметры для выравнивания?

Пусть у нас есть выравнивание. Если последовательности случайные и независимые (модель R), то вероятность увидеть букву α против β

$$p(\alpha, \beta \mid R) = p(\alpha) p(\beta)$$

а вероятность выравнивания (x, y) будет равна

$$p(x, y \mid R) = \prod p(x_i) \prod p(y_i)$$

Если выравнивание не случайно (модель M), то

$$p(x, y \mid M) = \prod p(x_i, y_i)$$

Отношение правдоподобия:

$$\frac{p(x, y \mid M)}{p(x, y \mid R)} = \frac{\prod p(x_i, y_i)}{\prod p(x_i) \prod p(y_i)}$$

Логарифмируя, получаем

$$\log(p(x, y \mid M) / p(x, y \mid R)) = \sum s(x_i, y_i);$$

<u>Матрица замен:</u> $s(\alpha, \beta) = \log(p_{\alpha\beta} / p_{\alpha} p_{\beta})$

Серия матриц BLOSUM

База данных BLOCKS (*Henikoff & Henikoff*) – безделеционные фрагменты множественных выравниваний (выравнивания получены экспертом).

В каждом блоке отбираем подмножество последовательностей, имеющих процент идентичных аминокислот не больше заданного значения ID.

В урезанном блоке в каждой колонке подсчитываем число пар аминокислот

$$n_{col}^{bl}(\alpha, \beta)$$

Усредняем по всем колонкам и по всем блокам:

$$f(\alpha, \beta) = \sum n_{col}^{bl}(\alpha, \beta) / N_{col}$$

Элемент матрицы BLOSUM_{ID}:

$$\mathbf{BLOSUM}_{ID}(\alpha, \beta) = \log(f(\alpha, \beta) / f(\alpha) f(\beta))$$

Серия матриц РАМ

Point Accepted Mutation – эволюционное расстояние, при котором произошла одна замена на 100 остатков.

Эволюционный процесс можно представить как Марковский процесс. Если в начальный момент времени $t=0$ в некоторой позиции был остаток α , то через время Δt в этой позиции с некоторой вероятностью будет остаток β :

$$p(\beta | \alpha, \Delta t) = M_{\Delta t}(\beta, \alpha)$$

M_{Δ} – эволюционная матрица

Через время $2 \cdot \Delta t$

$$p(\beta | \alpha, 2 \cdot \Delta t) = \sum_{\gamma} M_{\Delta t}(\beta, \gamma) \cdot M_{\Delta t}(\gamma, \alpha) = M_{\Delta t}^2(\beta, \alpha)$$

Через время $N \cdot \Delta t$

$$p(\beta | \alpha, N \cdot \Delta t) = M_{\Delta t}^N(\beta, \alpha)$$

Серия матриц РАМ

Находим выравнивания, отвечающие расстоянию РАМ1

Находим частоты пар и вычисляем частоты пар:

$$p(\alpha\beta) = p(\alpha \rightarrow \beta) p(\alpha) + p(\beta \rightarrow \alpha) p(\beta)$$

полагая $p(\alpha \rightarrow \beta) = p(\beta \rightarrow \alpha)$ получаем

$$p(\alpha \rightarrow \beta) = p(\alpha\beta) / (p(\alpha) + p(\beta))$$

$$p(\alpha \rightarrow \alpha) = 1 - \sum_{\beta \neq \alpha} p(\alpha \rightarrow \beta)$$

$$\text{РАМ}_N(\alpha\beta) = \log (p_{(\alpha \rightarrow \beta)}^N / p_\alpha p_\beta)$$

Локальное выравнивание

- Локальным оптимальным выравниванием называется такое оптимальное выравнивание фрагментов последовательностей, при котором любое удлинение или укорочение фрагментов приводит только к уменьшению веса.
- Локальному оптимальному выравниванию отвечает путь с наибольшим весом, независимо от того, где он начинается и где кончается.

Глобальное выравнивание

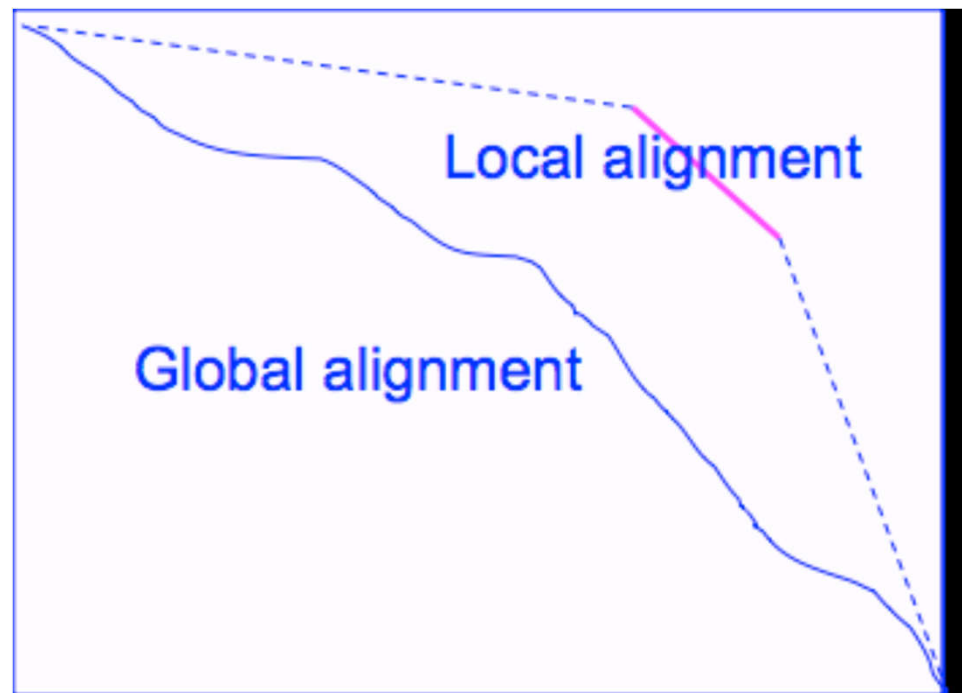
```
--T--CC-C-AGT--TATGT-CAGGGGACACG-A-GCATGCAGA-GAC
|  | | | |  | | | | |  | | | | |  |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG-T-CAGAT--C
```

Локальное выравнивание

```
          tccCAGTTATGTCAGgggacacgagcatgcagagac
          |||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

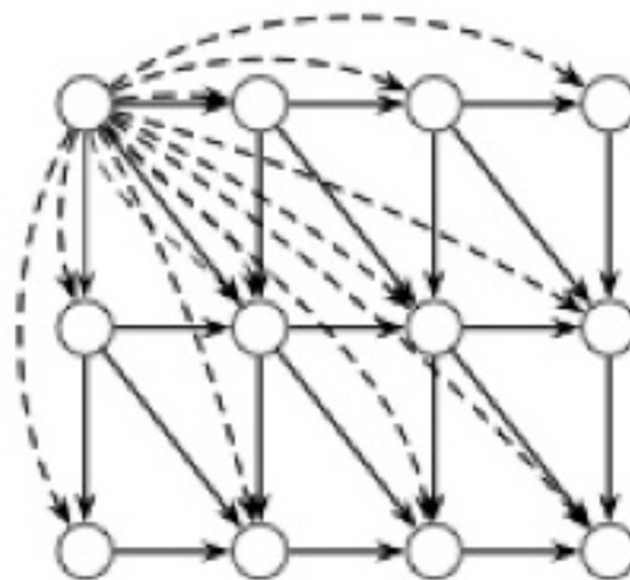
Локальное выравнивание

- Задача глобального выравнивания – найти наиболее весомый путь между вершинами $(0,0)$ и (n,m) графа.
- Задача локального выравнивания – найти наиболее весомый путь среди всех путей между вершинами (i,j) и (i',j') .



Алгоритм Смита-Ватермана

$$s_{i,j} = \max \left\{ \begin{array}{l} 0 \\ s_{i-1,j-1} + m(v_i, w_j) \\ s_{i-1,j} + m(v_i, -) \\ s_{i,j-1} + m(-, w_j) \end{array} \right.$$



Наибольшее значение $s_{i,j}$ —лучший вес локального выравнивания. Обратный проход начинается от максимального элемента в матрице и идем до нуля.

Аффинная функция штрафов

Удаление X букв подряд случается чаще, чем удаление X букв по отдельности.

Взвешивание делеций/вставок: простой подход.

- Фиксированный штраф σ за каждую делецию/вставку:

- $-\sigma$ за одну делецию,
- -2σ за две делеции подряд,
- -3σ за три делеции подряд, и т.д.

Аффинная функция штрафов

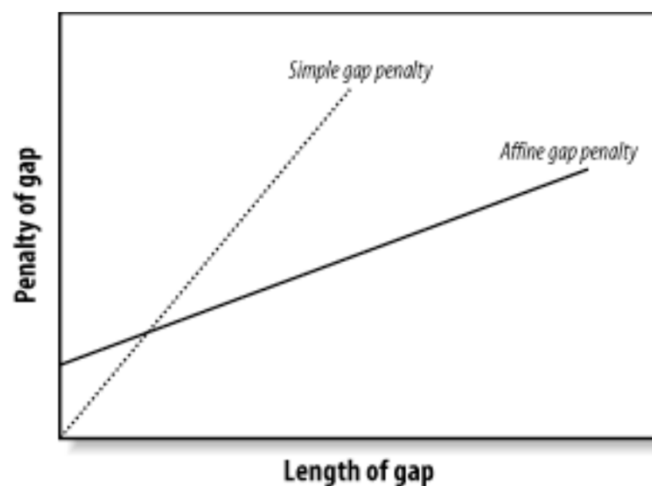
- В природе, серии последовательных k делеций происходят чаще, чем k одиночных событий:



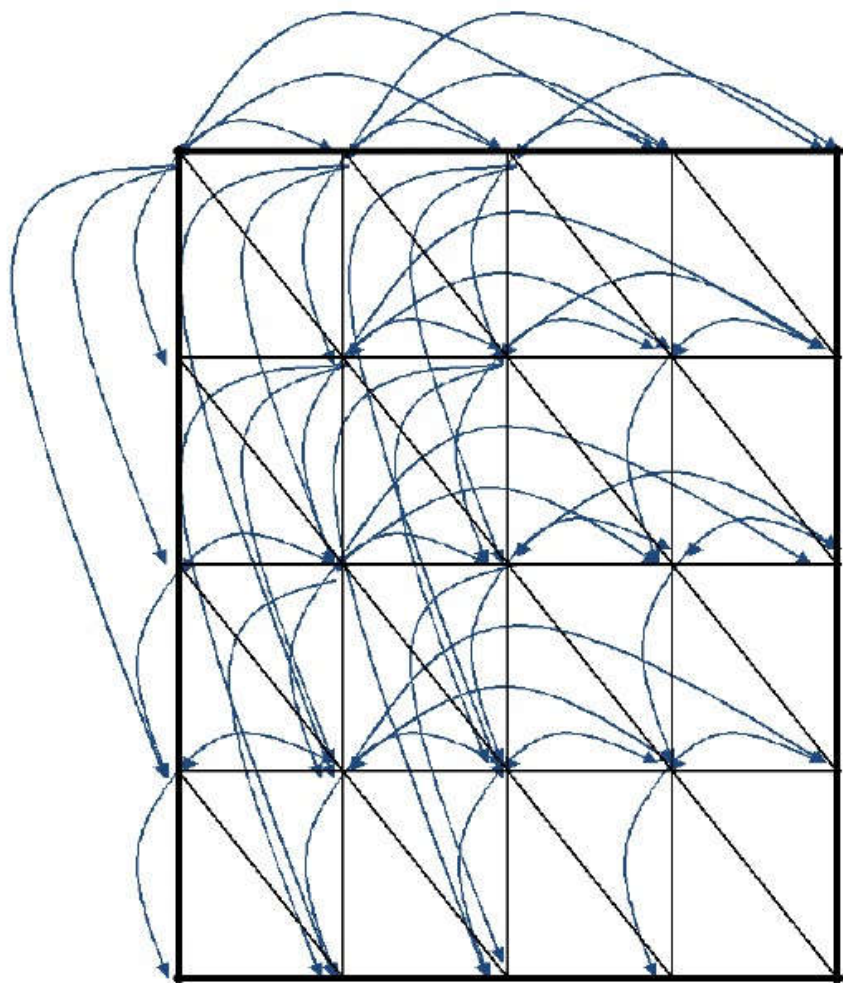
Аффинная функция штрафов

Аффинный штраф за гэпы

- $-\rho - \sigma$ за одну делецию 1indel
- $-\rho - 2\sigma$ за две делеции 2indels
- $-\rho - 3\sigma$ за три делеции 3indels, etc.

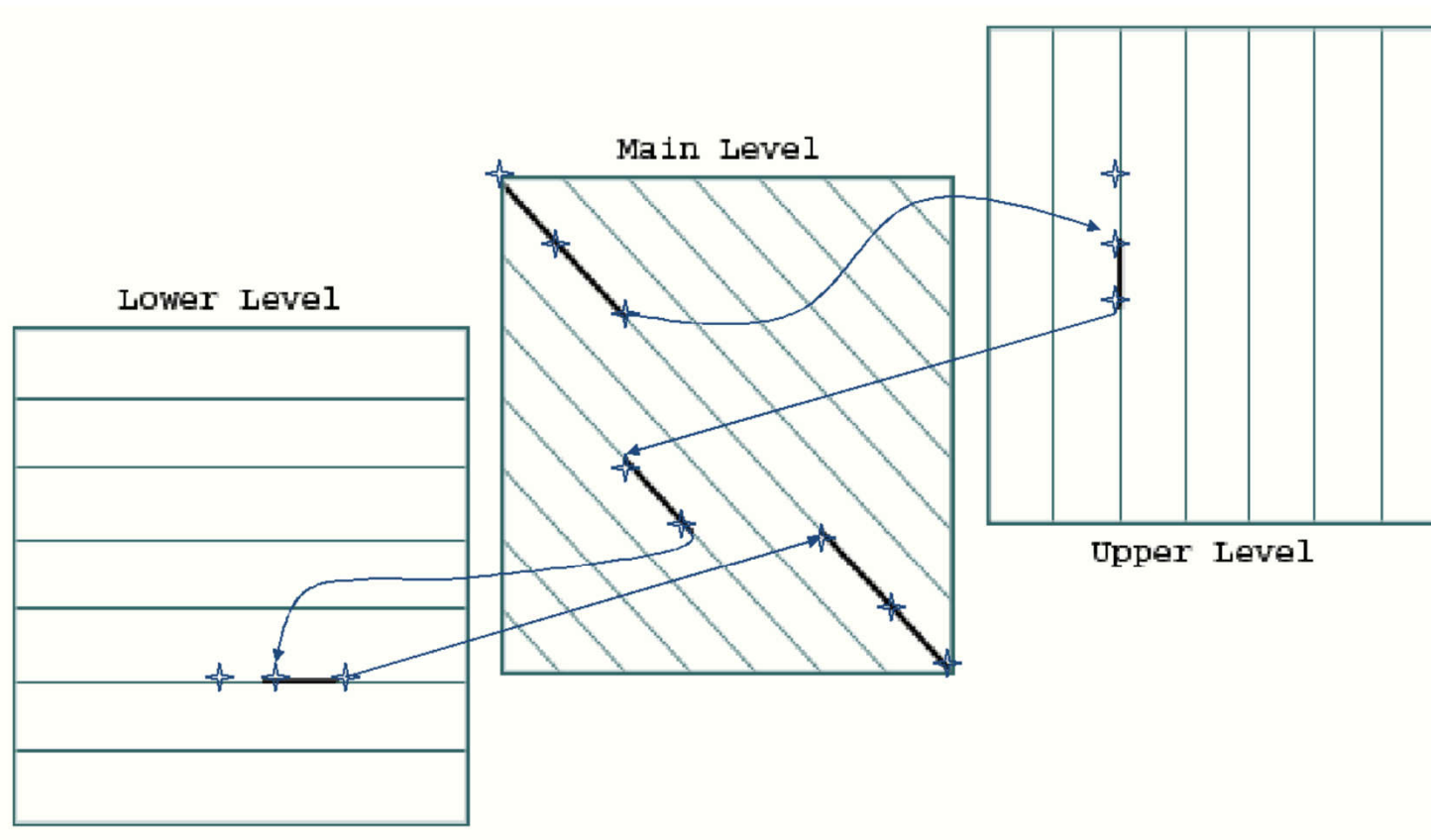


Аффинная функция штрафов



Добавление ребер аффинных
штрафов
Время работы возрастает до
 $O(n^3)$

Аффинная функция штрафов



Переключение между уровнями

- Уровни:

- Основной уровень для диагональных ребер
- Нижний уровень для горизонтальных ребер
- Верхний уровень для вертикальных ребер

- Штраф за переход с основного уровня на верхний или нижний (с шагом) ($-\rho-\sigma$)

- Штраф за проход по верхнему или нижнему уровню ($-\sigma$)

Алгоритм 3-х уровневого подхода

$$\downarrow s_{i,j} = \max \begin{cases} \downarrow s_{i-1,j} - \sigma \\ s_{i-1,j} - (\rho + \sigma) \end{cases}$$

Продолжит гэл в w (делеция)

Начать гэл в w (делеция): с середины

$$\rightarrow s_{i,j} = \max \begin{cases} \rightarrow s_{i,j-1} - \sigma \\ s_{i,j-1} - (\rho + \sigma) \end{cases}$$

Продолжить гэл в v (вставка)

Начать гэл в v (вставка): с середины

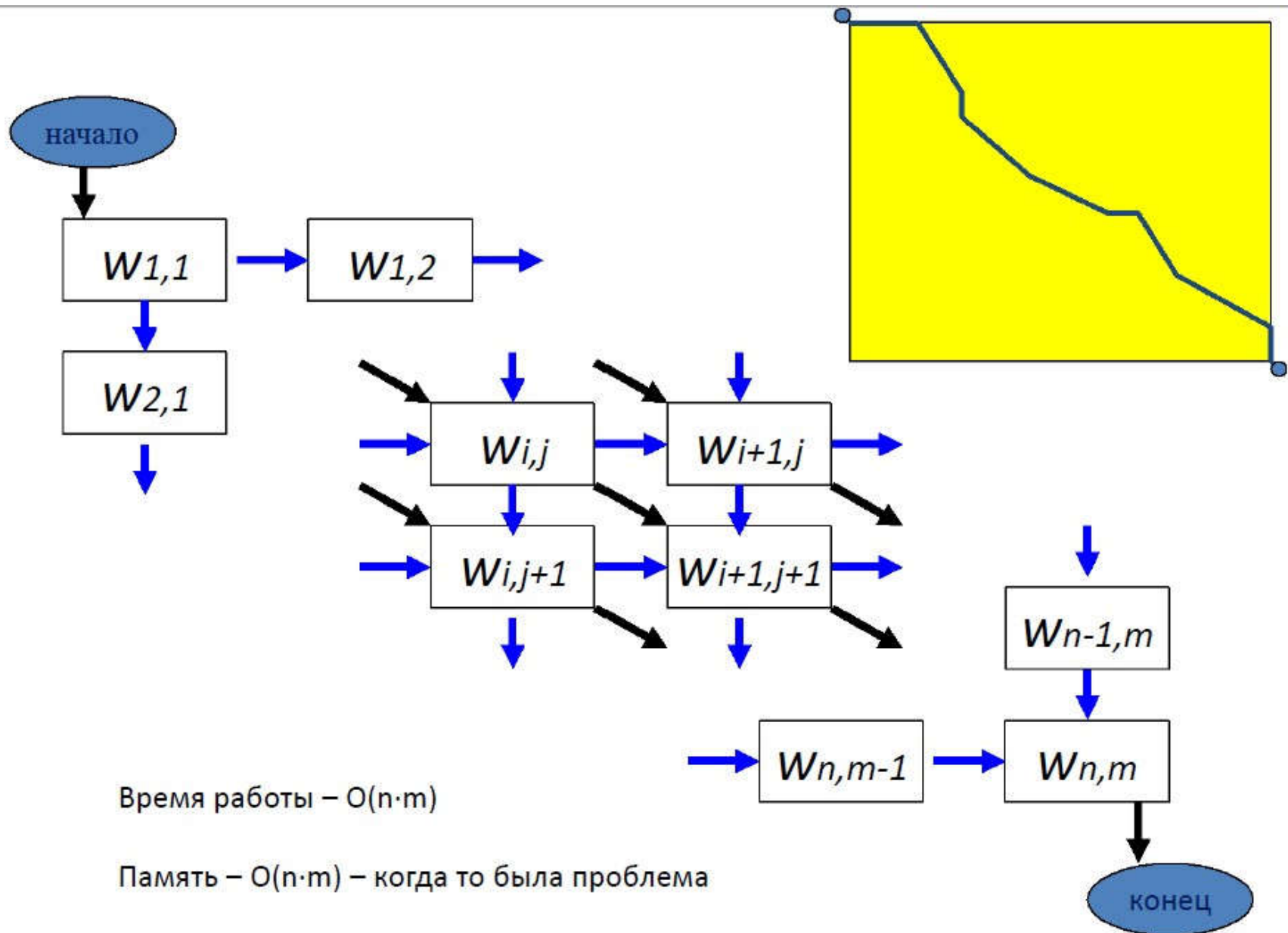
$$\swarrow s_{i,j} = \max \begin{cases} s_{i-1,j-1} + \delta(v_i, w_j) \\ s_{i,j} \downarrow \\ s_{i,j} \rightarrow \end{cases}$$

Совпадение или несовпадение

Закончить делецию: сверху

Закончить вставку: снизу

Алгоритм 3-х уровневого подхода



Программные средства попарного выравнивания

EMBOSS programs -

needle - global alignment

water - local alignment

FASTA programs -

align - global alignment

lalign - local alignment

BLAST programs -

bl2seq - local alignment

EBI: <http://www.ebi.ac.uk/Tools/psa/> .

Имеется описание всех используемых инструментов,
наиболее используемые

Needle (global alignment):

http://www.ebi.ac.uk/Tools/psa/emboss_needle/help/index-protein.html

Water (local alignment):

http://www.ebi.ac.uk/Tools/psa/emboss_water/help/index-protein.html

Программные средства попарного выравнивания

BL2seq - local alignment

[https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE_TYPE=BlastSearch
&PROG_DEF=blastn&BLAST_PROG_DEF=megaBlast&BLAST_SPEC=blast2seq](https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE_TYPE=BlastSearch&PROG_DEF=blastn&BLAST_PROG_DEF=megaBlast&BLAST_SPEC=blast2seq)

FASTA - LALIGN

https://fasta.bioch.virginia.edu/fasta_www2/fasta_www.cgi
https://embnet.vital-it.ch/software/LALIGN_form.html

Программные средства попарного выравнивания

```
#
# Aligned_sequences: 2
# 1: EMBOSS_001
# 2: EMBOSS_002
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 361
# Identity:   176/361 (48.8%)
# Similarity: 214/361 (59.3%)
# Gaps:       92/361 (25.5%)
# Score: 860.5
#
#
#=====
```

Пример выравнивания Программой EBI Needle

```
EMBOSS_001      1 ----- 0
EMBOSS_002      1 MRQSLKVMVLSTVALLFMANPAAASEEKKEYLIVVEPEEVSAQSVVEESYD 50
EMBOSS_001      1 -----AQSVPWGI 8
                               :|:|||||
EMBOSS_002     51 VDIHEFEFEEIPVIHAELTKKELKKLKKDPNVKAIEKNAEVTISQTPWGI 100
EMBOSS_001      9 SRVQAPAAHNRGLTGSGVKVAVLDTGISTHPDLNIRGGASFVPGEPTQD 58
   |:....:||||:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|
EMBOSS_002    101 SFINTQQAHNRGIFGNGARVAVLDTGIASHPDLRIAGGASFISSEPSYHD 150
EMBOSS_001     59 GNGHGHVAGTIAALNNSIGVLGVAPSAELYAVKVLGASGSGSVSSIAQG 108
   .|||||:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|
EMBOSS_002    151 NNGHGHVAGTIAALNNSIGVLGVAPSADLYAVKVLDRNGSGSLASVAQG 200
EMBOSS_001    109 LEWAGNNGMHVANLSLGSPPSATLEQAVNSATSRGVLVVAASGNSGAGS 158
   :|||.|||.||:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|
EMBOSS_002    201 IEWAINNNMHIIINMSLGSTSGSSTLELAVNRANNAGILLVGAAGNTGRQG 250
```

Множественное выравнивание

- Обобщение парного выравнивания
- Выравнивание 2-х последовательностей –
- двумерная матрица
- 3-х последовательностей –3-х мерная.

A T _ G C G _

A _ C G T _ A

A T C A C _ A

- Задача: больше консервативных столбцов, лучше
выравнивание

Множественное выравнивание

- Перенос аннотации
- Предсказание функции каждого остатка (например, выявление остатков, составляющих активный центр фермента)
- Моделирование 3D – структуры
- Реконструкция эволюционной истории последовательности (филогения)
- Выявление паттерна функциональных семейств и сигналов в ДНК
- Построение доменных профайлов
- Аккуратный дизайн праймеров для PCR анализа

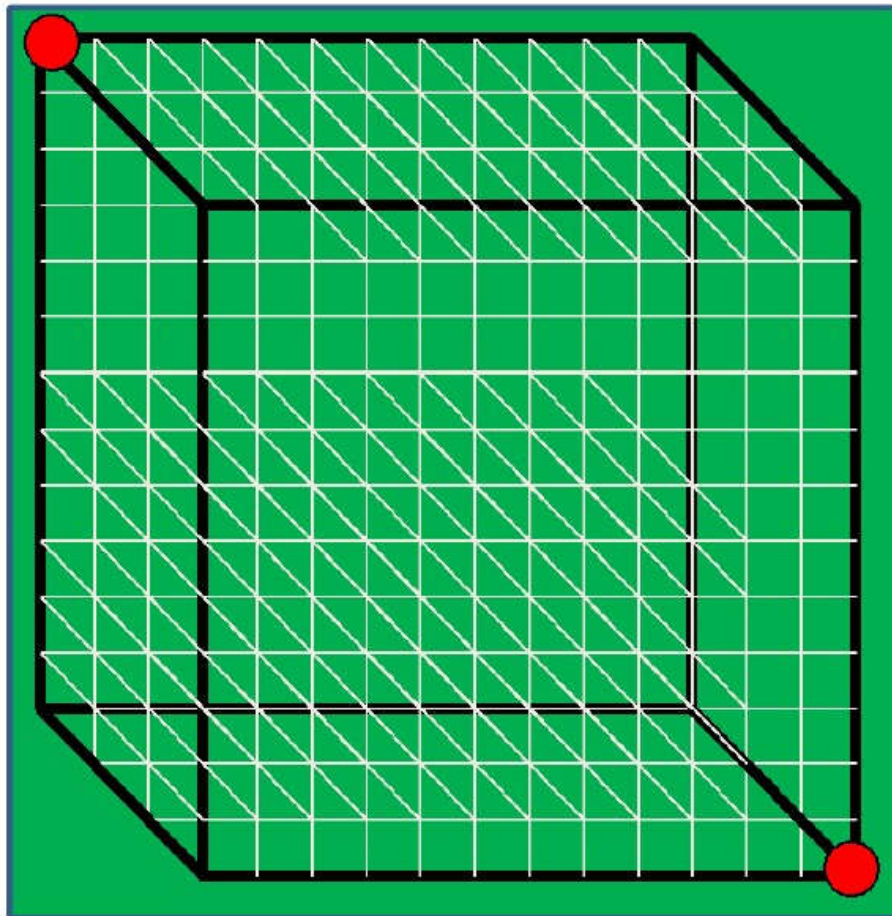
Множественное выравнивание (рекомендации)

- Выравнивайте белки, а не ДНК, если есть выбор
- Последовательностей лучше много, но не слишком ($\sim 10-15$)
- В выборке лучше избегать:
 - слишком похожих последовательностей ($>90\%$ id)
 - слишком разных последовательностей ($<30\%$ id с большинством)
 - неполных последовательностей (фрагментов)
 - тандемных повторов

Множественное выравнивание

- Глобальное выравнивание 3-х последовательностей

начало

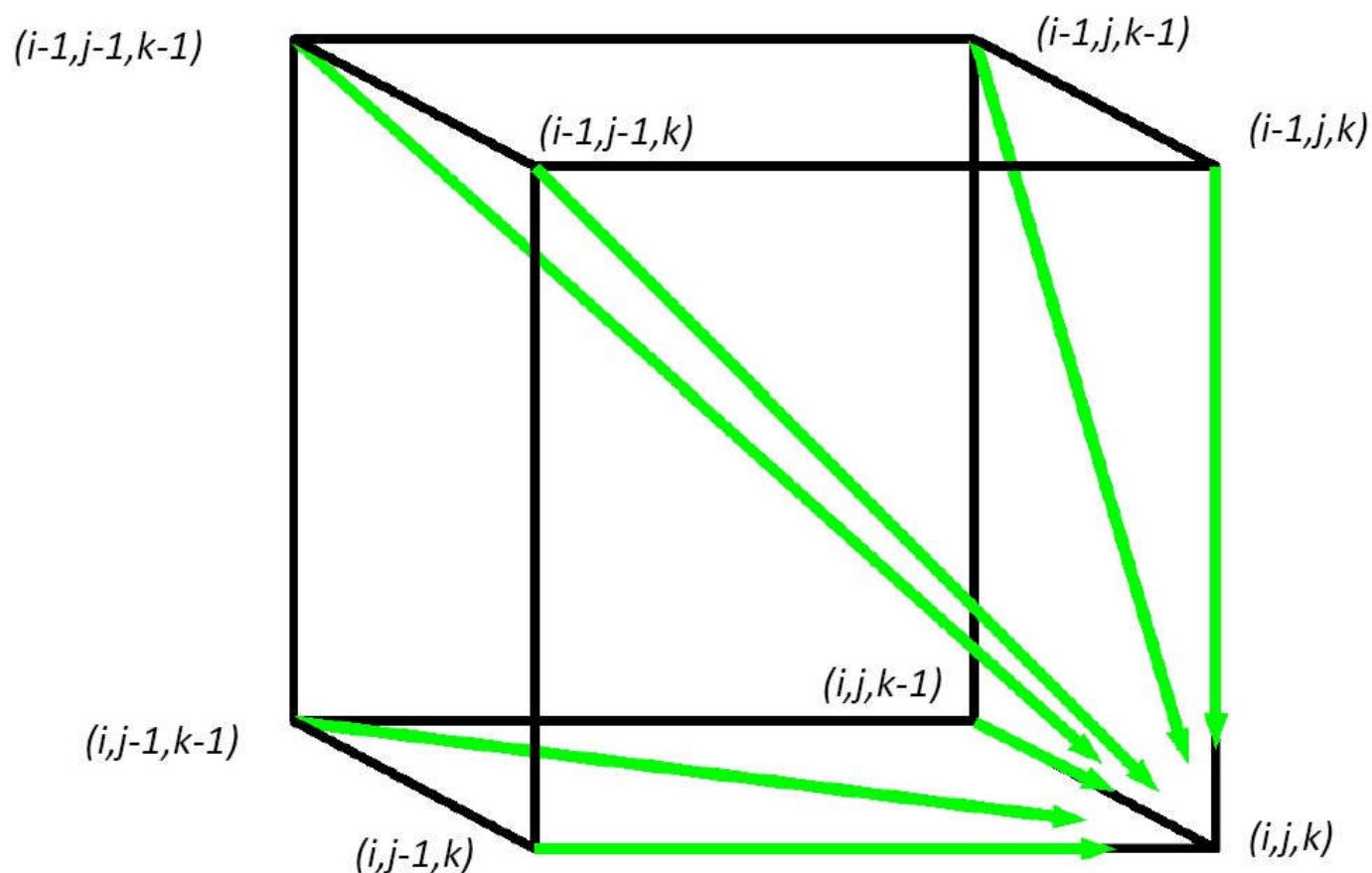


конец

Множественное выравнивание

- Глобальное выравнивание 3-х последовательностей

3-D архитектура



Множественное выравнивание

- Глобальное выравнивание 3-х последовательностей

Алгоритм

- $s_{i,j,k} = \max \left\{ \begin{array}{ll} s_{i-1,j-1,k-1} + \delta(v_i, w_j, u_k) & \text{Нет гэпов} \\ s_{i-1,j-1,k} + \delta(v_i, w_j, _) & \text{Один гэп} \\ s_{i-1,j,k-1} + \delta(v_i, _, u_k) & \text{Один гэп} \\ s_{i,j-1,k-1} + \delta(_, w_j, u_k) & \text{Один гэп} \\ s_{i-1,j,k} + \delta(v_i, _, _) & \text{Два гэпа} \\ s_{i,j-1,k} + \delta(_, w_j, _) & \text{Два гэпа} \\ s_{i,j,k-1} + \delta(_, _, u_k) & \text{Два гэпа} \end{array} \right.$

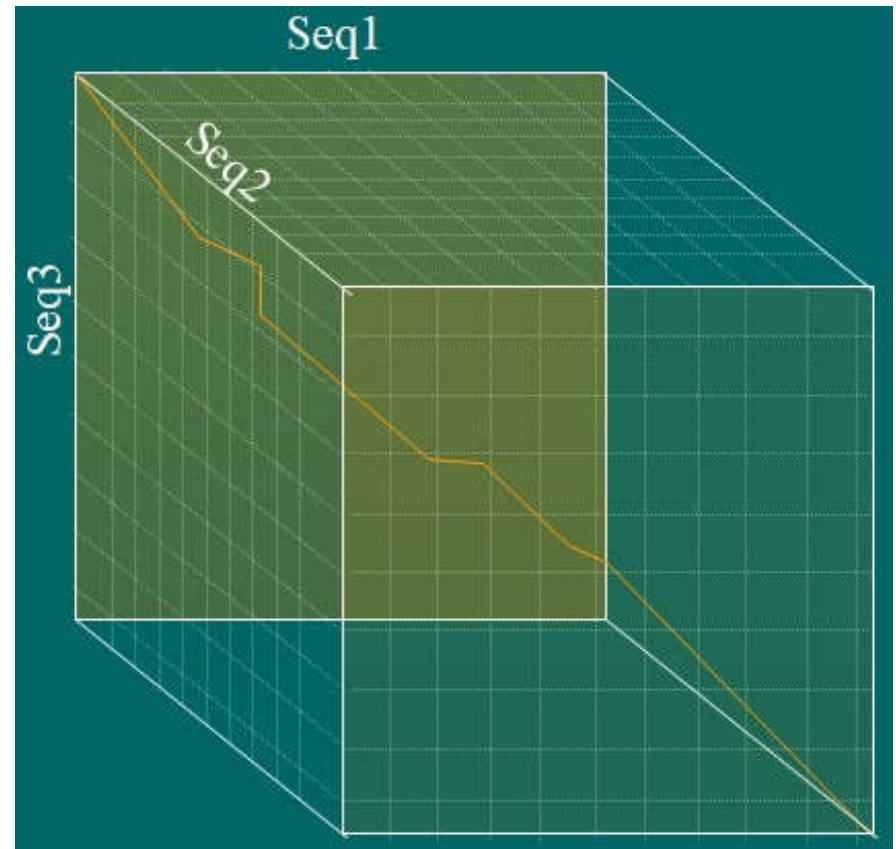
- $\delta(x, y, z)$ – запись в трехмерной матрице весов

Множественное выравнивание

- Глобальное выравнивание 3-х последовательностей

Время работы алгоритма

- Для 3-х последовательностей длины n , время работы $-7n^3$; $O(n^3)$
- Для k последовательностей - $(2k-1)(n^k)$; $O(2kn^k)$



Если количество последовательностей > 4 , то задача практически не разрешима.

Множественное выравнивание

Множественное выравнивание порождает парные выравнивания

x: AC-GCGG-C
y: AC-GC-GAG
z: GCCGC-GAG

Порождает:

x: ACGCGG-C ;	x: AC-GCGG-C ;	y: AC-GCGAG
y: ACGC-GAC ;	z: GCCGC-GAG ;	z: GCCGCGAG

Множественное выравнивание

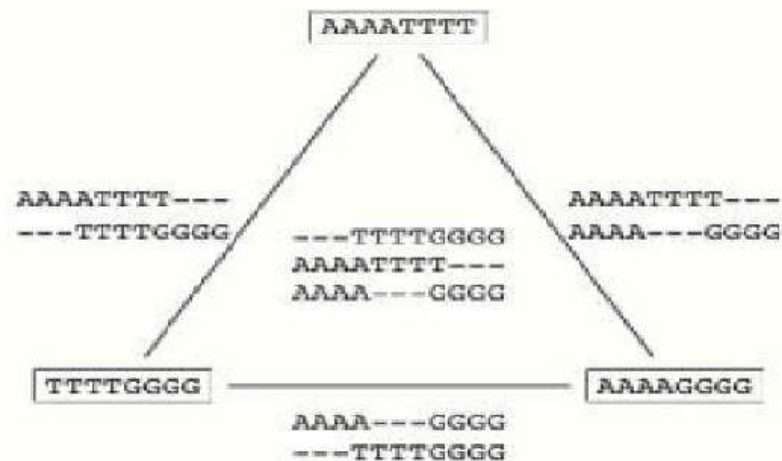
Обратная проблема

Имея 3 субъективных парных выравнивания:

x:	ACGCGG-C;	x:	AC-GCGG-C;	y:	AC-GCGAG
y:	ACGC-GAC;	z:	GCCGC-GAG;	z:	GCCGCGAG

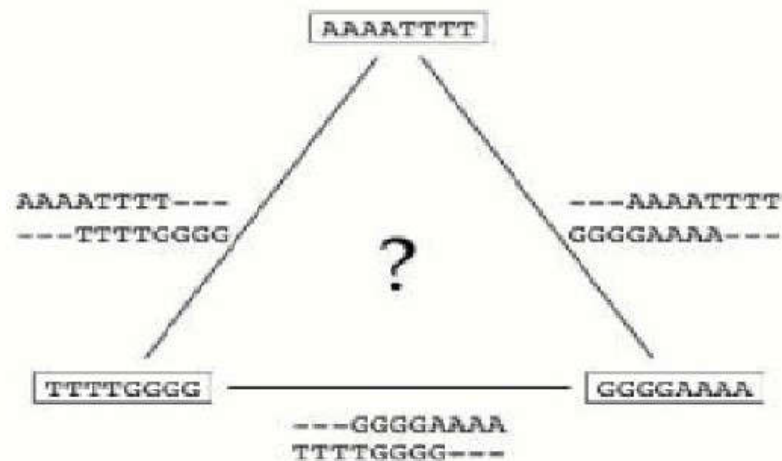
Множественное выравнивание

Хороший вариант



(a) Compatible pairwise alignments

Плохой вариант



(b) Incompatible pairwise alignments

Множественное выравнивание

Описание выравнивания

$\left. \begin{array}{l} \text{GTC} \text{ TGA} \\ \text{GTC} \text{ AGC} \end{array} \right\} \begin{array}{l} \text{GTC} [\text{TA}] \text{G} [\text{AC}] \\ \text{G} [5\text{X}] \\ [6\text{X}] \end{array} \quad - \text{ профиль}$

x GGGCACTGCAT

y GGTTACGTC--

z GGGAACCTGCAG

w GGACGTACC--

v GGACCT-----

GGACACAGCAT - консенсус

Множественное выравнивание – жадный алгоритм

$$k \left\{ \begin{array}{l} u_1 = \text{ACGTACGTACGT}\dots \\ u_2 = \text{TTAATTAATTA}\dots \\ u_3 = \text{ACTACTACTACT}\dots \\ \dots \\ u_k = \text{CCGGCCGGCCGG} \end{array} \right. \begin{array}{l} \longrightarrow \\ \nearrow \\ \end{array} \left. \begin{array}{l} u_1 = \text{ACg/tTACg/tTACg/cT}\dots \\ u_2 = \text{TTAATTAATTA}\dots \\ \dots \\ u_k = \text{CCGGCCGGCCGG} \end{array} \right\} k-1$$

Время работы алгоритма на k последовательностях длины n – $O(n^2k^2)$

Множественное выравнивание – жадный алгоритм

Рассмотрим 4 последовательности: GATTCA, GTCTGA, GATATT, GTCAGC

Имеется 6 возможных парных выравниваний:

s_2 GTCTGA
 s_4 GTCAGC (score = 2)

s_1 GATTCA--
 s_4 G-T-CAGC (score = 0)

s_1 GAT-TCA
 s_2 G-TCTGA (score = 1)

s_2 G-TCTGA
 s_3 GATAT-T (score = -1)

s_1 GAT-TCA
 s_3 GATAT-T (score = 1)

s_3 GAT-ATT
 s_4 G-TCAGC (score = -1)

Множественное выравнивание – жадный алгоритм

s_2 and s_4 самые похожие, соединим две последовательности с использованием матрицы профиля:

$$\left. \begin{array}{l} s_2 \text{ GTC} \textcolor{red}{TGA} \\ s_4 \text{ GTC} \textcolor{red}{AGC} \end{array} \right\} s_{2,4} = \text{GTC} \textcolor{red}{t/aGa/c}$$

Имеем новое множество из 3 последовательностей для выравнивания:

$$\begin{array}{l} s_1 \quad \text{GATTCA} \\ s_3 \quad \text{GATATT} \\ s_{2,4} \quad \text{GTC} \textcolor{red}{t/aGa/c} \end{array}$$

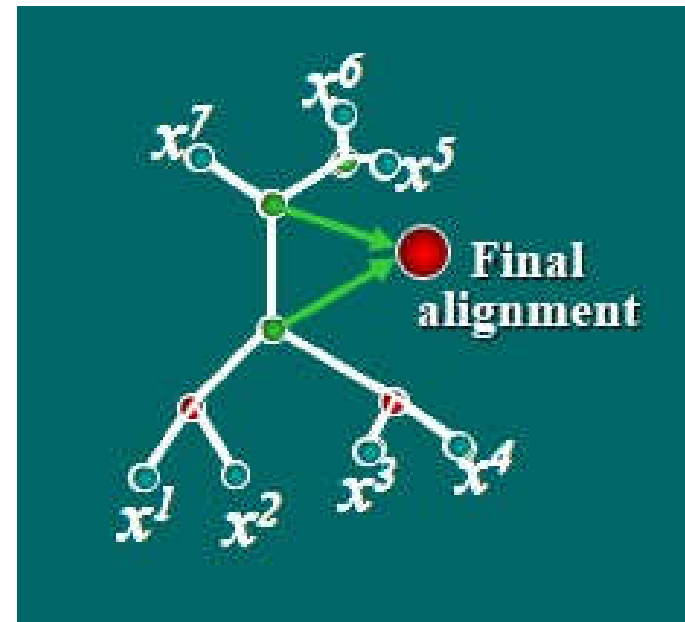
Можем выбрать любой из нуклеотидов для $s_{2,4}$.

Множественное выравнивание – Прогрессивное выравнивание

Строится бинарное дерево (guide tree, путеводное дерево) – листья = последовательности

Дерево обходится начиная с листьев. При объединении двух узлов строится парное выравнивание супер-последовательностей (профилей) и получается новая суперпоследовательность

Путеводное дерево строится приближенно – главное быстро. Обычно это кластерное дерево



Множественное выравнивание – ClustalW

Прогрессивное выравнивание – жадный алгоритм с более «умным» способом выбора пар.

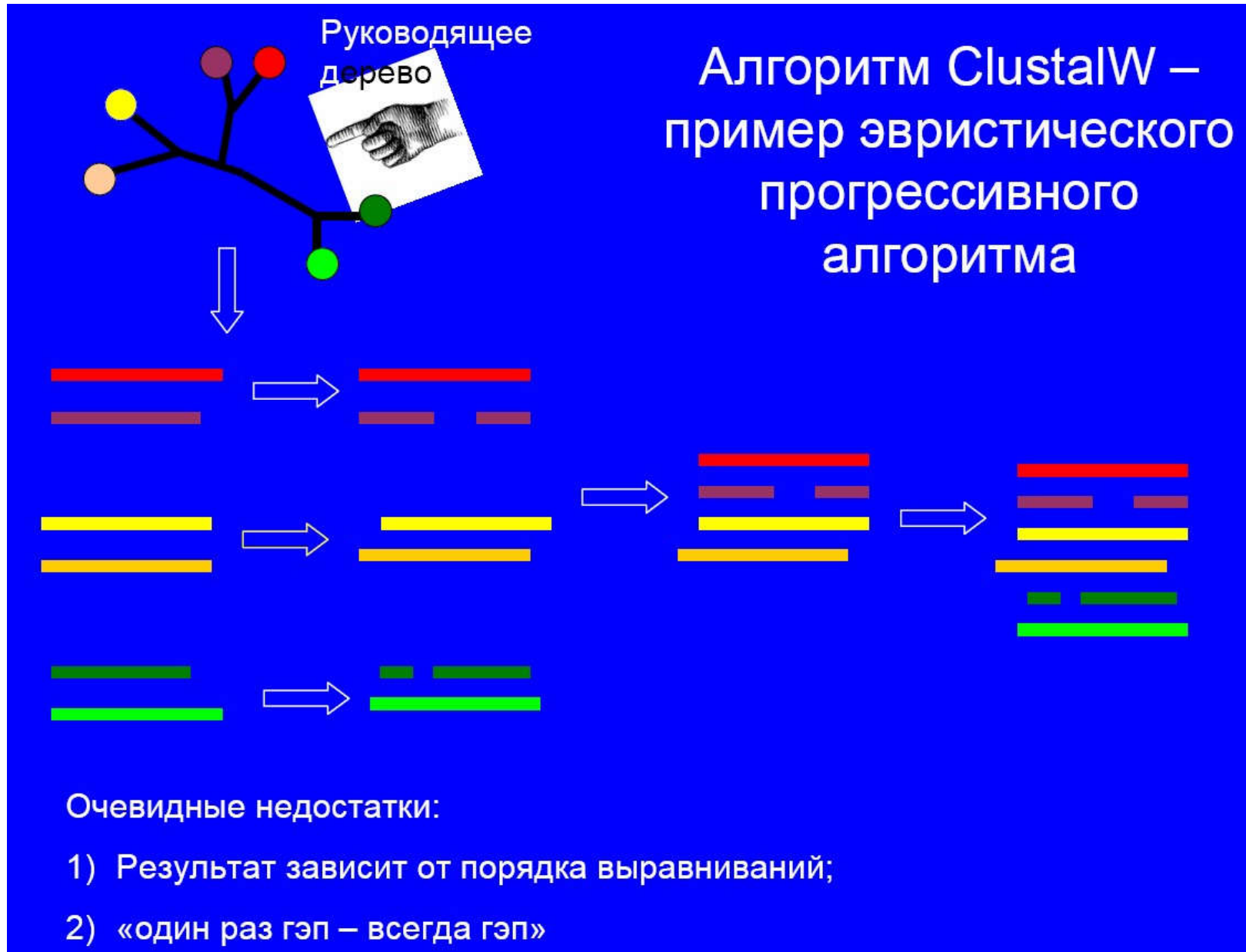
- Три шага

- 1.) Построить парные выравнивания

- 2.) Построить дерево-подсказку

- 3.) Прогрессивное выравнивание по дереву-подсказке

Множественное выравнивание – ClustalW



Множественное выравнивание – ClustalW

Шаг 1: Парные Выравнивания

- Выравнивания пар порождают матрицу identity

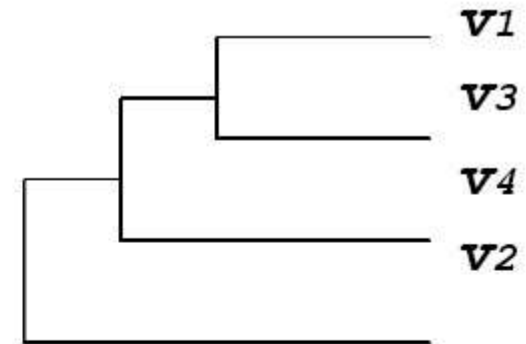
	v1	v2	v3	v4
v1	—			
v2	.17	—		
v3	.87	.28	—	
v4	.59	.33	.62	—

(.17 значит идентичны на 17 %)

Множественное выравнивание – ClustalW

Шаг 2: Дерево-подсказка

	v1	v2	v3	v4
v1	-			
v2	.17	-		
v3	.87	.28	-	
v4	.59	.33	.62	-



Далее вычислить:

v1,3 = выравнивание (v1, v3)

v1,3,4 = выравнивание ((v1,3), v4)

v1,2,3,4 = выравнивание ((v1,3,4), v2)

Множественное выравнивание – ClustalW


Шаг 3: Прогрессивное выравнивание

Выравниванием 2 наиболее близких последовательности.

- Следуя дереву -подсказке, довыравниваем следующую последовательность к имеющемуся выравниванию.

FOS_RAT PEEMSVTS-LDLTGGLPEATTPESEEAFTLPLLNDPEPK-PSLEPVKNISNMELKAEFPD
FOS_MOUSE PEEMSVAS-LDLTGGLPEASTPESEEAFTLPLLNDPEPK-PSLEPVKSISNVELKAEFPD
FOS_CHICK SEELAAATALDLG----APSPAAAEAAAFALPLMTEAPPAVPPKEPSG--SGLELKAEPFD
FOSB_MOUSE PGPGPLAEVRDLPG-----STSAKEDGFGWLLPPPPPPP-----LFPQ
FOSB_HUMAN PGPGPLAEVRDLPG-----SAPAKEDGFSWLLPPPPPPP-----LFPQ

. . : ** . :.. *:.* * . * **:



Точки и звезды отображают насколько консервативны столбцы.

Современные методы построения множественного выравнивания (MSA, multiple sequence alignment):

Алгоритм ClustalW (реализации **ClustalX**, **emma** из EMBOSS) – до сих пор самый популярный, но уже устаревший метод (на Web – например, <https://www.ebi.ac.uk/Tools/msa/clustalo/>)

Muscle – быстрее и немного точнее, самый новый и довольно модный (<https://www.ebi.ac.uk/Tools/msa/muscle/>)

T-COFFEE – заметно точнее, но существенно медленнее (<https://www.ebi.ac.uk/Tools/msa/tcoffee/>)

Множественное выравнивание

Как “читать” множественное выравнивание?

Хорошее выравнивание — высоко- консервативные блоки, перемежающиеся блоками с инсерциями/делециями

ДНК — консервативные “островки”

Качество — score, локально важно

“consensus” — строка с символами “*”, “:”, “.” — консервативный, похожие по размеру и гидропатичности, похожие по размеру ИЛИ гидропатичности, соответственно

Улучшение выравнивания

Недостаток прогрессивных методов: если для некоторой группы последовательностей выравнивание построено, то оно уже не перестраивается.

Алгоритм итеративного улучшения

1. Вынимаем из выравнивания одну последовательность
2. По оставшимся последовательностям строим профиль
3. Выравниваем вынутую последовательность с профилем
4. Переходим к этапу 1.

Множественное выравнивание

Множественные Выравнивания:

Взвешивание

- Количество полных совпадений
- Сумма по парам(SP-Score)
- Энтропия

Множественное выравнивание

Количество полных совпадений

AАА
AАА
AАТ
АТС

- Хорошо только для очень близких последовательностей

Множественное выравнивание

|Сумма по парам (SP-Score)

- Построим парное выравнивание по множественному
- Посчитаем веса всех этих парных выравниваний — $s(a_i, a_j)$
- Просуммируем: $s(a_1, \dots, a_k) = \sum_{i,j} s(a_i, a_j)$

Множественное выравнивание

Энтропия

- Определим вероятности букв в столбцах
 - $p_A = 1, p_T = p_G = p_C = 0$ (1-ый столбец)
 - $p_A = 0.75, p_T = 0.25, p_G = p_C = 0$ (2-ый столбец)
 - $p_A = 0.50, p_T = 0.25, p_C = 0.25, p_G = 0$ (3-ий столбец)
- Энтропия столбца будет равна

$$- \sum_{X=A,T,G,C} p_X \log p_X$$

AAA
AAA
AAT
ATC

Множественное выравнивание

Энтропия: Пример

Лучший вариант $entropy \begin{pmatrix} A \\ A \\ A \\ A \end{pmatrix} = 0$

Худший вариант $entropy \begin{pmatrix} A \\ T \\ G \\ C \end{pmatrix} = -\sum \frac{1}{4} \log \frac{1}{4} = -4(\frac{1}{4} * -2) = 2$

Множественное выравнивание

Энтропия: Пример

Энтропия столбца:

$$-(p_A \log p_A + p_C \log p_C + p_G \log p_G + p_T \log p_T)$$

A	A	A
A	C	C
A	C	G
A	C	T

- Столбец 1 = $-[1 * \log(1) + 0 * \log 0 + 0 * \log 0 + 0 * \log 0]$
= 0

- Столбец 2 = $-[(1/4) * \log(1/4) + (3/4) * \log(3/4) + 0 * \log 0 + 0 * \log 0]$
= $-[(1/4) * (-2) + (3/4) * (-.415)] = +0.811$

- Столбец 3 = $-[(1/4) * \log(1/4) + (1/4) * \log(1/4) + (1/4) * \log(1/4) + (1/4) * \log(1/4)]$
= $4 * -[(1/4) * (-2)] = +2.0$

- Энтропия выравнивания = $0 + 0.811 + 2.0 = +2.811$