Множественное выравнивание кодирующих последовательностей с учётом сдвигов рамки считывания

Студент: Батусов П. В.
Руководитель: Страшнов П. В.

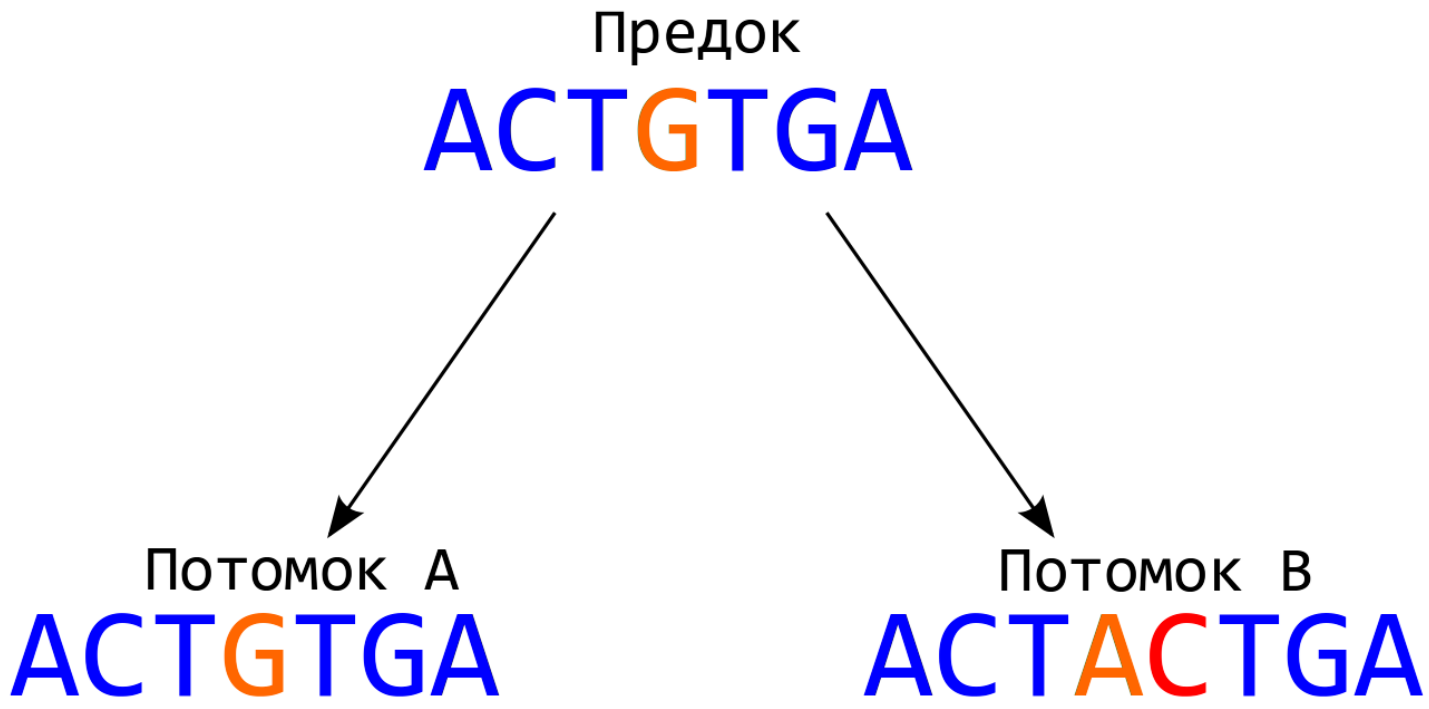
Постановка задачи

Цель работы – создание приложения для построения множественного выравнивания кодирующих последовательностей нуклеотидов с учетом их трансляции на аминокислотном уровне

Задача выравнивания

Выравнивание аминокислотных или нуклеотидных последовательностей – это процесс сопоставления сравниваемых последовательностей для такого их взаиморасположения, при котором наблюдается максимальное количество совпадений аминокислотных остатков или нуклеотидов

Поиск гомологий в биологических последовательностях



Задача выравнивания

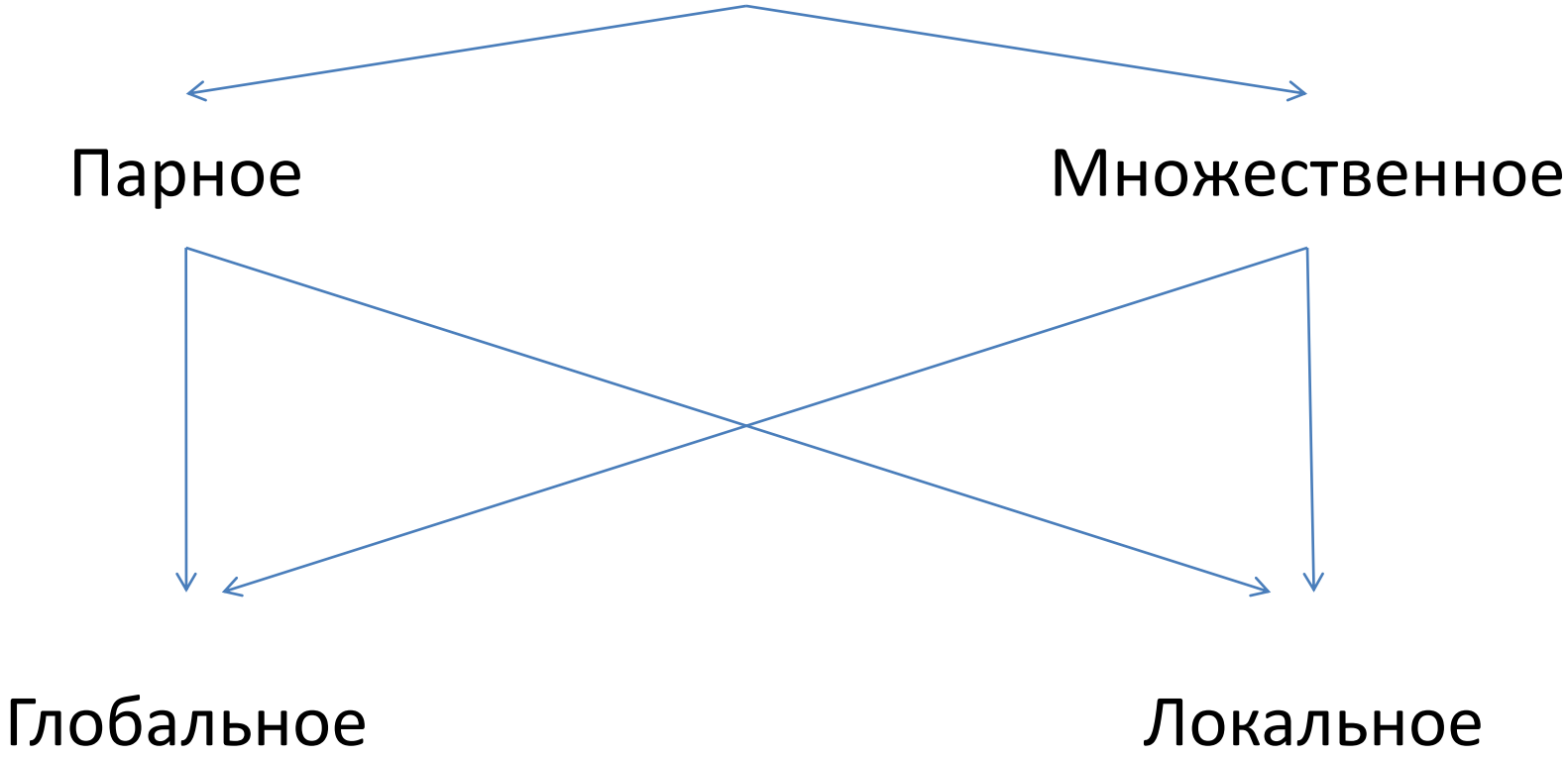
ДНК предка: …AAACTGATGCAACGTGA…

ДНК потомка: …AATCTTTGATACCTGA…

Выравнивание – общепринятый способ отражения “родства” двух последовательностей нуклеотидов или аминокислотных остатков

```
Seq1:   …AAAC--TGATGCAACGTGA…
Seq2:   …AATCTTTGAT---ACCTGA…
```

Выравнивание



Классические методы поиска гомологий

Алгоритм Нидлмана-Вунша

- Строит глобальное выравнивание двух последовательностей
- $O(len(S_1) \cdot len(S_2))$

Алгоритм Смита-Ватермана

- Строит локальное выравнивание двух последовательностей
- $O(len(S_1) \cdot len(S_2))$

Алгоритм Нидлмана-Вунша

- S(a,b) – похожесть символов a и b
- Линейный штраф за разрыв d

	А	**Г**	**Т**	**Ц**
А	10	−1	−4	−3
Г	−1	7	−3	−5
Т	−4	−3	8	0
Ц	−3	−5	0	9

Базис:

- $F_{0,j} = d \cdot j$
- $F_{i,0} = d \cdot i$

Итерационная формула:

$$F_{i,j} = max \begin{cases} F_{i-1,j-1} + S_{A_i,B_j} \\ F_{i-1,j} + d \\ F_{i,j-1} + d \end{cases}$$

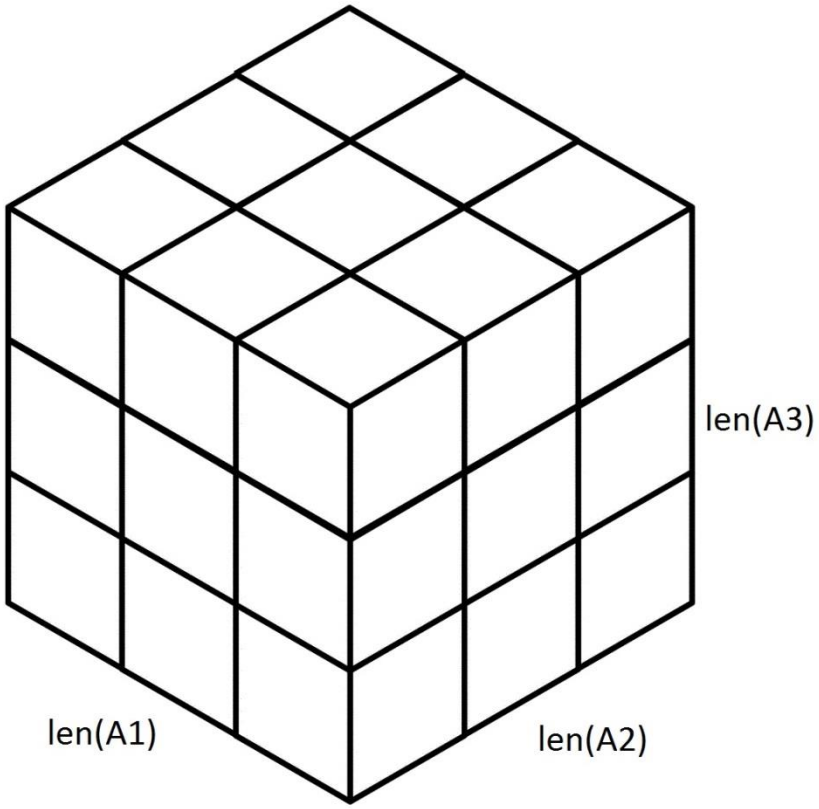
Алгоритм Смита-Ватермана

$$F_{i,j} = max \begin{cases} F_{i-1,j-1} + S_{A_i,B_j} \\ F_{i-1,j} + D_{A_i} \\ F_{i,j-1} + I_{B_j} \\ 0 \end{cases}$$

Множественное выравнивание
Выравнивание в кубе

Сложность алгоритма для n-мерного случая:

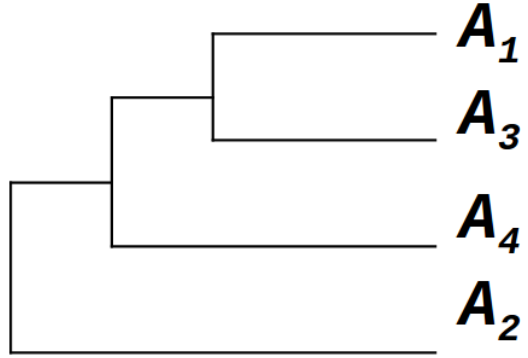
$$O((2^n - 1)\prod_{i=1}^{n} len(A_i))$$



Выравнивание выравниваний
Алгоритм Clustal

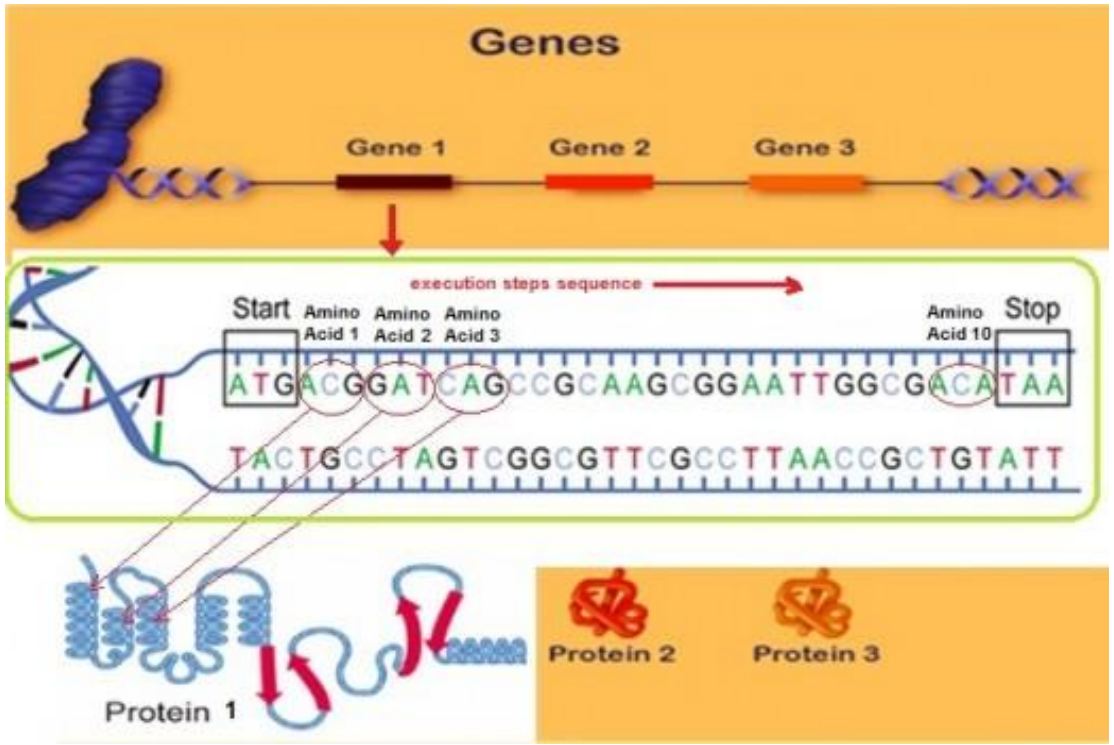
$$f(f(f(\ldots f(f(A_1, A_2), A_3)\ldots), A_{n-1}), A_n)$$

	A_1	A_2	A_3	A_4
A_1	-			
A_2	.17	-		
A_3	.87	.28	-	
A_4	.59	.33	.62	-



Открытые рамки считывания

Открытая рамка считывания (ОРС) - нуклеотидная последовательность нуклеиновой кислоты (ДНК или РНК), потенциально способная кодировать белок.



Трёхступенчатый подход

Идея алгоритма

- Трансляция исходной последовательности нуклеотидов по всем возможным рамкам считывания
- Выравнивание последовательности аминокислот «классическими» алгоритмами
- Трансляция полученного белка обратно в последовательность нуклеотидов

Проблемы

- Алгоритм не учитывает возможные изменения рамки считывания
- Невозможность расширения до задачи множественного выравнивания

Двухуровневое выравнивание

Идея алгоритма

- штраф за выравнивание является сочетанием двух штрафов: на аминокислотном и нуклеотидном уровнях
- инсерции допустимы только на аминокислотном уровне (запрет на сдвиг рамки считывания)

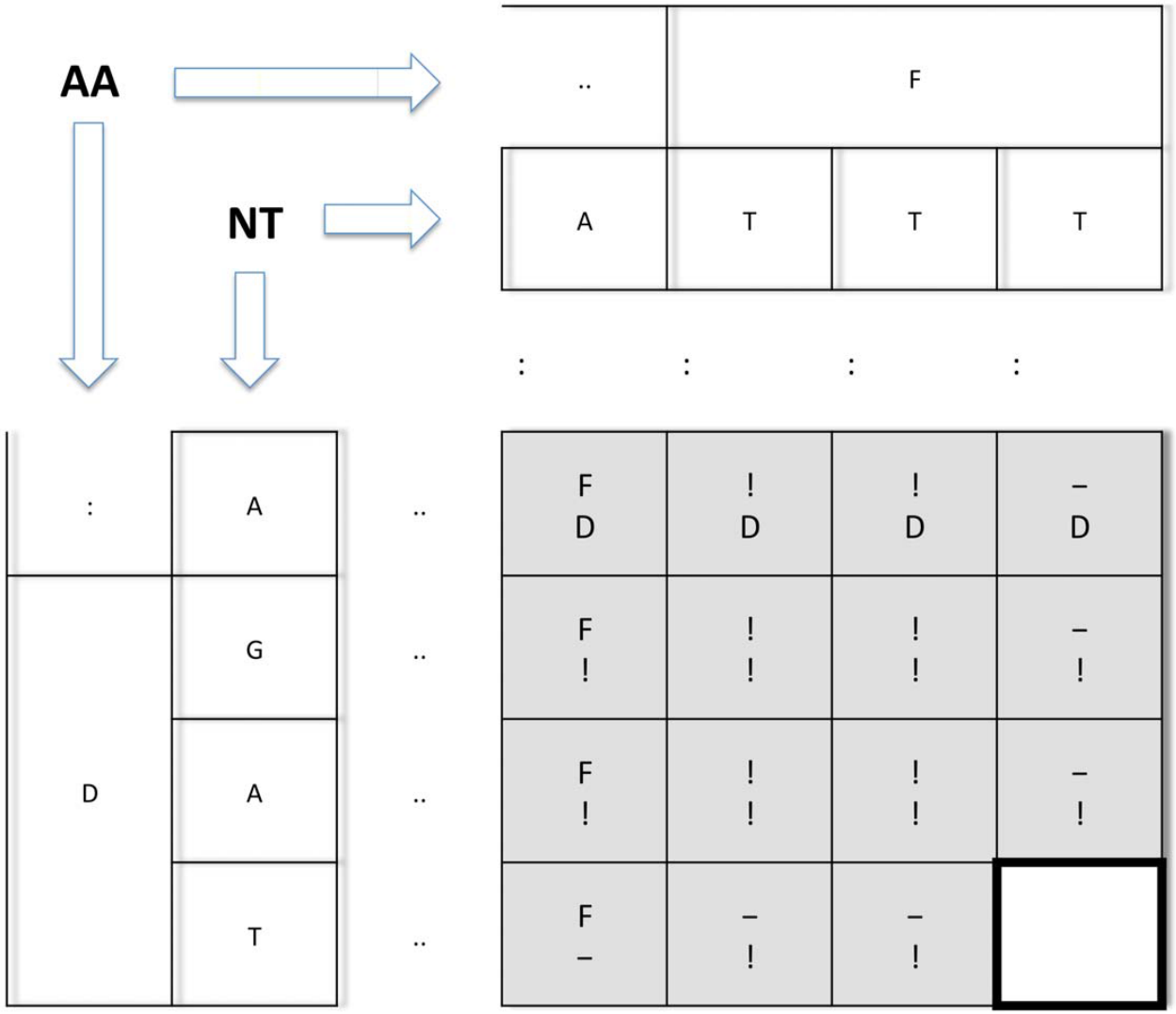
Проблемы

- Высокая вычислительная сложность

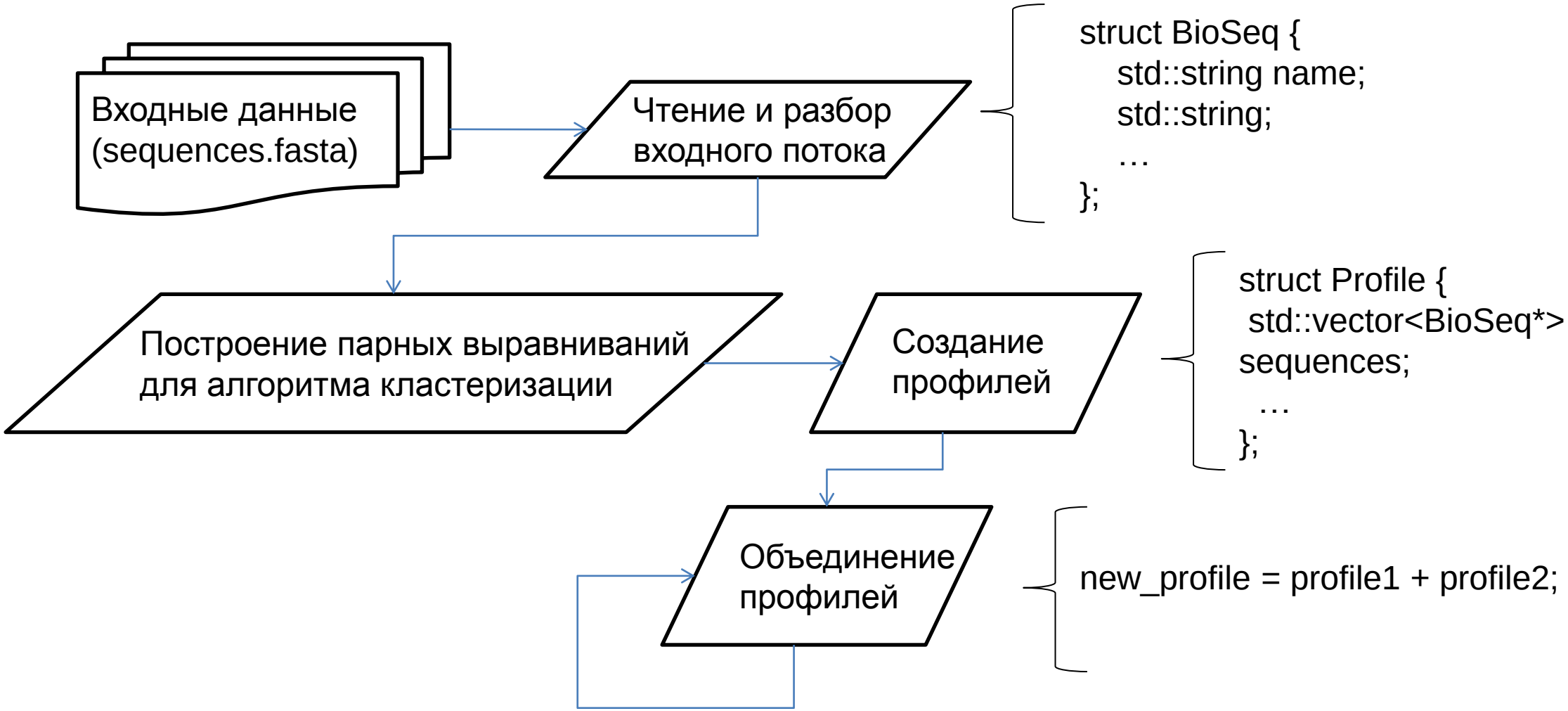
MACSE

Алгоритм основан на идее двухуровневого выравнивания, но имеет меньшую вычислительную сложность и позволяет строить множественные выравнивания, с учетом открытых рамок считывания.

MACSE производит выравнивание выравниваний, выбирая порядок через дерево-подсказку, как и алгоритм Clustal.



Общая схема работы алгоритма



Чтение и разбор входного потока

Структура для хранения последовательности

```
struct BioSeq {
    std::string name;
    std::string;
   …
};
```

Входные данные

Последовательности в формате FASTA:

> Идентификатор #1
последовательность 1
последовательность 1
…
> Идентификатор #2
последовательность 2
…

Алгоритм кластеризации UPGMA

1. Перед началом работы алгоритма рассчитывается матрица расстояний между объектами. Каждый объект образует свой собственный кластер.
2. Ищется минимальное значение, соответствующее расстоянию между двумя наиболее близкими кластерами. Найденные кластеры объединяются, образуя новый кластер.
3. Расстояние между кластерами *(u,v)* и *w* определяется согласно формуле:

$$D\big((u,v),w\big) = \frac{T_u D_{u,w} + T_v D_{v,w}}{T_u + T_v}$$

	1	2	3	4	5
1	0	2.06	4.03	6.32	2.08
2	2.06	0	3.50	4.12	5.43
3	4.03	3.50	0	2.25	3.65
4	6.32	4.12	2.25	0	4.81
5	2.08	5.43	3.65	4.81	0

Объединение профилей

Количество нуклеотидов \ Разрыв рамки	Последний геп у второго триплета	Последний геп у первого триплета	Без гепа в конце триплета
1	--X ---	--- --X	--X --X
2	-XX -X-	-X- -XX	-XX -XX
	-XX ---	--- -XX	-XX --X
			--X -XX
3	XXX XX-	XX- XXX	XXX XXX
	XXX -X-	-X- XXX	XXX -XX
	XXX X--	X-- XXX	-XX XXX
	XXX ---	--- XXX	XXX X-X
			XXX --X
			X-X XXX
			--X XXX

```
struct Profile {
 std::vector<BioSeq*>
sequences;
  …
};
```

$$new\_profile = profile1 + profile2$$