

Выравнивания. Продолжение.

Алгоритмы в биоинформатике

Антон Елисеев

eliseevantoncoon@gmail.com

Что было на прошлой лекции?

- Дали определение выравнивания и веса выравнивания.
- Узнали что замены неравноценны.
- Обсудили как устроены матрицы замен BLOSUM и PAM.
- Обсудили что один длинный гэп более вероятен чем много коротких.
- Использовали субаддитивные функции штрафов за гэпы, в частности линейную.

Что будет на этой лекции?

- Узнаем про локальные выравнивания.
- Обсудим затраты памяти на выравнивание, научимся экономить.
- Поговорим про идеи альтернативных алгоритмов выравнивания.

Локальное выравнивание

Зачем?

Локальное выравнивание

$S_{global}(a, b) = \max_{(a^*, b^*)} W(a^*, b^*)$ - лучшее глобальное выравнивание

$S_{local}(a, b) = \max_{(i_l, i_r), (j_l, j_r)} S_{global}(a[i_l \dots i_r], b[j_l \dots j_r])$ локальное выравнивание

Локальное выравнивание

$S_{global}(a, b) = \max_{(a^*, b^*)} W(a^*, b^*)$ - лучшее глобальное выравнивание

$S_{local}(a, b) = \max_{(i_l, i_r), (j_l, j_r)} S_{global}(a[i_l \dots i_r], b[j_l \dots j_r])$ локальное выравнивание

Замечание

Будем веса выбирать так, чтобы пенализировать удаления, вставки и мутации, а за совпадения буреом поощрять.

Локальное выравнивание

Идея:

$$D_{i,j} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \text{ (замена)} \\ D_{i-1,j} + w(a_i, _) \text{ (удаление)} \\ D_{i,j-1} + w(_, b_j) \text{ (вставка)} \end{cases}$$

Локальное выравнивание

Идея:

Если так вышло, что ячейка в динамике стала отрицательной, то нам не нужно ее продолжать! Просто начнём сначала.

$$D_{i,j} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_i) \text{ (замена)} \\ D_{i-1,j} + w(a_i, _) \text{ (удаление)} \\ D_{i,j-1} + w(_, b_j) \text{ (вставка)} \end{cases}$$

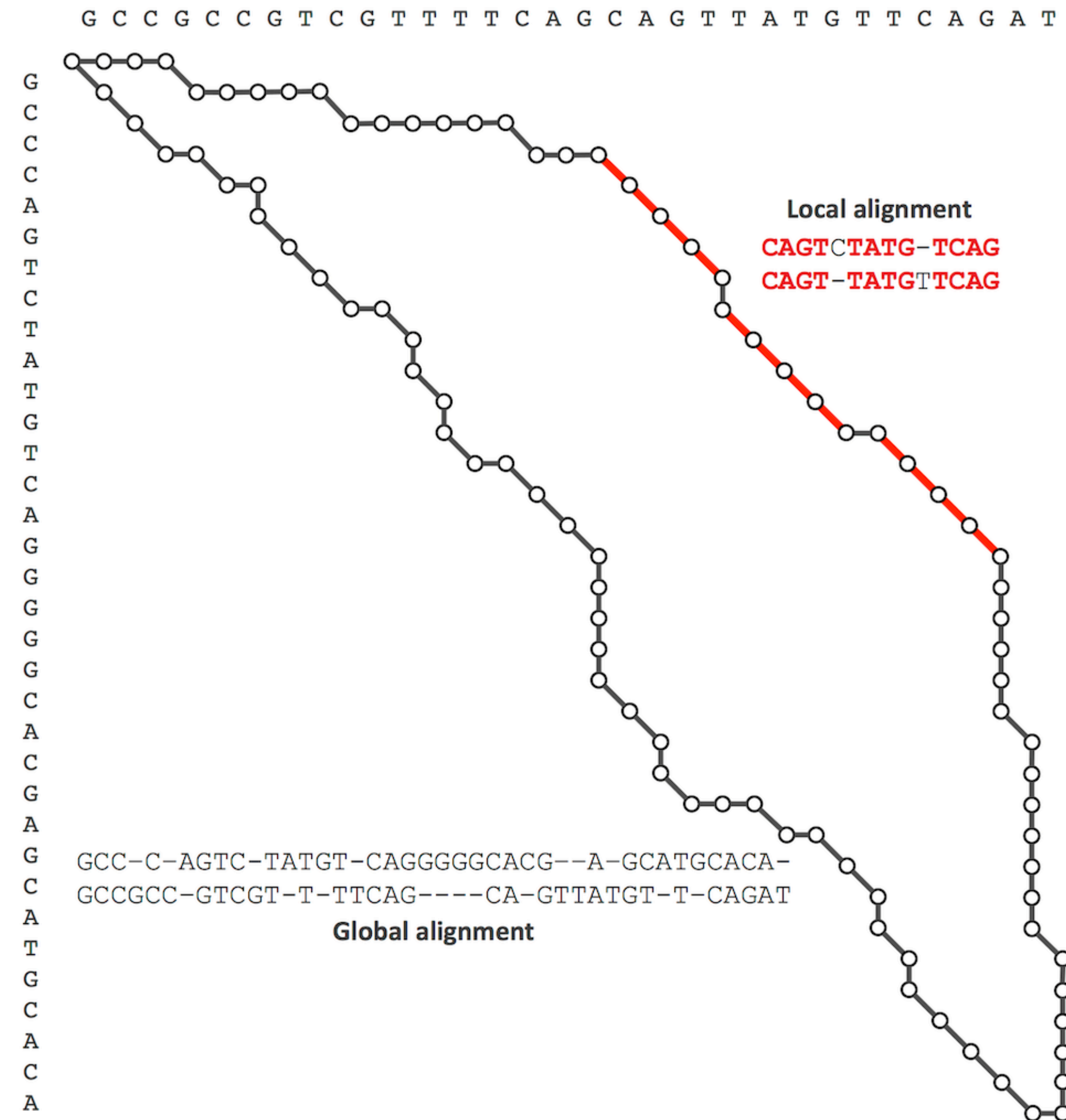
Алгоритм Смита с Ватерманом.

- $D_{0,0} = 0$
- $D_{i,0} = D_{i-1,0} + w(a_i, _)$
- $D_{0,j} = D_{0,j-1} + w(_, b_j)$
- $$D_{i,j} = \max \begin{cases} 0 \text{ (пустое выравнивание)} \\ D_{i-1,j-1} + w(a_i, b_j) \text{ (замена)} \\ D_{i-1,j} + w(a_i, _) \text{ (удаление)} \\ D_{i,j-1} + w(_, b_j) \text{ (вставка)} \end{cases}$$

Локальное выравнивание

Как найти само выравнивание?

Алгоритм Смита с Ватерманом.



Локальное выравнивание

- Сложность по времени

Локальное выравнивание

- Сложность по времени
 $O(n^2)$
- Сложность по памяти

Локальное выравнивание

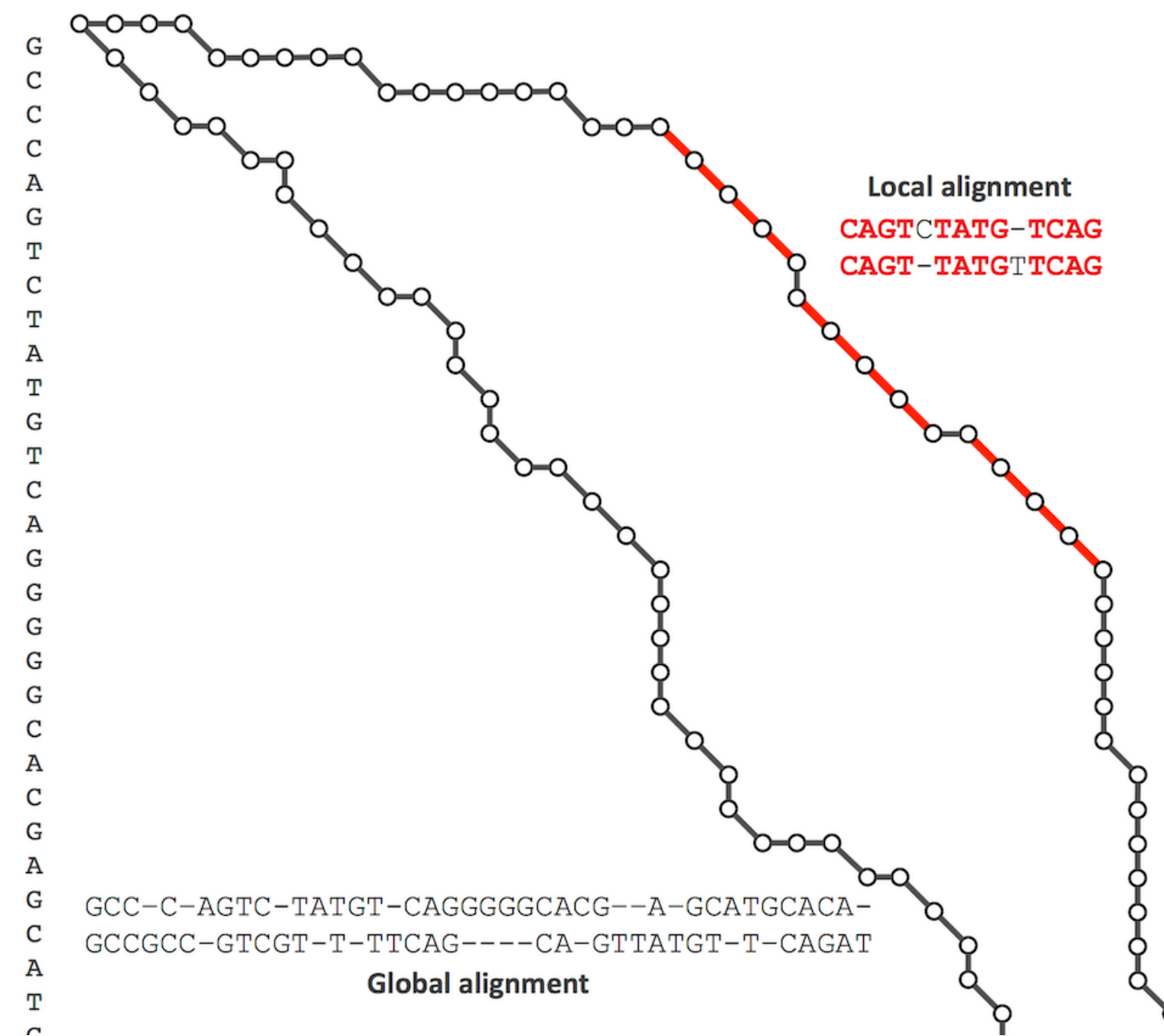
- Сложность по времени
 $O(n^2)$
- Сложность по памяти
 $O(n^2)$

Локальное выравнивание. Замечания.

- Работает так же быстро как глобальное
- Можно находить много локальных выравниваний!

Локальное выравнивание. Замечания.

- Работает так же быстро как глобальное
- Можно находить много локальных выравниваний!



Алгоритмы выравнивания и память

- Needleman–Wunsch
 $O(n^2)$
- Gotoh's
 $O(n^2)$
- Smith/Waterman
 $O(n^2)$

Алгоритмы выравнивания и память

Найдем выравнивание X и Y хромосом человека.

$$\text{size}(X) = 156040895 \sim 10^8$$

$$\text{size}(Y) = 57227415 \sim 10^7$$

Нам понадобится больше 10^{15} бит

Алгоритм Хиршберга!

Леммы:

- Пусть $D(a, b)$ расстояние выравнивания, тогда $D(a^{-1}, b^{-1}) = D(a, b)$, где x^{-1} последовательность x в обратном порядке.

Алгоритм Хиршберга!

Леммы:

- Пусть $D(a, b)$ расстояние выравнивания, тогда $D(a^{-1}, b^{-1}) = D(a, b)$, где x^{-1} последовательность x в обратном порядке.
- Пусть $D(a, b)$ — расстояние редактирования и $a = a_l + a_r$, где $a_l = a[..k]$, $a_r = a[k..]$. Тогда \exists такое i , что для $b_l = b[..i]$, $b_r = b[i..]$ выполняется $D(a, b) = D(a_l, b_l) + D(a_r, b_r)$

Алгоритм Хиршберга!

Пруф:

↯ некоторое разбиение $a = a_l + a_r$.

Допустим что $\nexists i$ такого, для
которого выполняется

$$D(a, b) = D(a_l, b_l) + D(a_r, b_r)$$

Алгоритм Хиршберга!

Пруф:

∄ некоторое разбиение $a = a_l + a_r$.

Допустим что $\nexists i$ такого, для которого выполняется

$$D(a, b) = D(a_l, b_l) + D(a_r, b_r)$$

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга!

Пруф:

∄ некоторое разбиение $a = a_l + a_r$.

Допустим что $\nexists i$ такого, для которого выполняется

$$D(a, b) = D(a_l, b_l) + D(a_r, b_r)$$

\Rightarrow если мы разобьем a на a_l и a_r , и выровняем a_l с любым из префиксов b , то a_r нельзя будет выровнять ни с одним из суффиксов b , так, чтобы получилось как в выравнивании a, b .

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга!

Замечание

Допустим что $\exists i$ такое, что выполняется

$$D(a, b) > D(a_l, b_l) + D(a_r, b_r)$$

Но тогда мы бы объединили выравнивание a_l, b_l и a_r, b_r и получили бы более оптимальное выравнивание!

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга!

Следствие

Получается что если $a = a_l + a_r$, то найдется такой индекс i , что для $b_l = b[..i]$, $b_r = b[i..]$ выполняется $D(a, b) = D(a_l, b_l) + D(a_r, b_r)$ но такого индекса где $D(a, b) > D(a_l, b_l) + D(a_r, b_r)$ точно не найдется!

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга!

Следствие

Получается что если $a = a_l + a_r$, то найдется такой индекс i , что для $b_l = b[..i]$, $b_r = b[i..]$ выполняется $D(a, b) = D(a_l, b_l) + D(a_r, b_r)$ но такого индекса где $D(a, b) > D(a_l, b_l) + D(a_r, b_r)$ точно не найдется.

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга!

Следствие

Получается что если $a = a_l + a_r$, то найдется такой индекс i , что для $b_l = b[..i]$, $b_r = b[i..]$ выполняется $D(a, b) = D(a_l, b_l) + D(a_r, b_r)$ но такого индекса где $D(a, b) > D(a_l, b_l) + D(a_r, b_r)$ точно не найдется.

Значит можно выравнивать суффикс и префикс и искать минимум!

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

$$a_l = GAT, \quad a_r = TACA$$

Алгоритм Хиршберга.

1. Разбиваем a пополам, $a = a_{\dots\frac{1}{2}} + a_{\frac{1}{2}\dots}$
2. Находим расстояние выравнивание от $a_{\dots\frac{1}{2}}$ до всех префиксов b и то же самое для $a_{\frac{1}{2}\dots}^{-1}$ и b^{-1}
3. Находим разбиение b такое, при котором минимально $D(a_{\dots\frac{1}{2}}, b_l) + D(a_{\frac{1}{2}\dots}, b_r)$, записываем, где это произошло.
4. Запускаемся рекурсивно на $a_{\dots\frac{1}{2}}, b_l$ и на $a_{\frac{1}{2}\dots}, b_r$

Алгоритм Хиршберга. Пример.

$a = AGTACGCA$
 $b = TATGC$

удаления и вставки — 2
совпадения 2
мутации — 1

$$H(AGTACGCA, TATGC)$$


$$NW(AGTA, b)$$

| | | | | | | |
|---|----|----|----|----|----|-----|
| | | T | A | T | G | C |
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | 0 | -2 | -4 | -6 |
| G | -4 | -3 | -2 | -1 | 0 | -2 |
| T | -6 | -2 | -4 | 0 | -2 | -1 |
| A | -8 | -4 | 0 | -2 | -1 | -3 |


$$NW(\text{rev}(CGCA), \text{rev}(b))$$

| | | | | | | |
|---|----|----|----|----|----|-----|
| | | C | G | T | A | T |
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | -3 | -5 | -4 | -6 |
| C | -4 | 0 | -2 | -4 | -6 | -5 |
| G | -6 | -2 | 2 | 0 | -2 | -4 |
| C | -8 | -4 | 0 | 1 | -1 | -3 |

Алгоритм Хиршберга. Пример.

$a = AGTACGCA$

$b = TATGC$

удаления и вставки -2

совпадения 2

мутации -1

$H(AGTACGCA, TATGC)$

$NW(AGTA, b)$

| | | T | A | T | G | C |
|---|----|----|----|----|----|-----|
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | 0 | -2 | -4 | -6 |
| G | -4 | -3 | -2 | -1 | 0 | -2 |
| T | -6 | -2 | -4 | 0 | -2 | -1 |
| A | -8 | -4 | 0 | -2 | -1 | -3 |

$NW(\text{rev}(CGCA), \text{rev}(b))$

| | | C | G | T | A | T |
|---|----|----|----|----|----|-----|
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | -3 | -5 | -4 | -6 |
| C | -4 | 0 | -2 | -4 | -6 | -5 |
| G | -6 | -2 | 2 | 0 | -2 | -4 |
| C | -8 | -4 | 0 | 1 | -1 | -3 |

Алгоритм Хиршберга. Пример.

$a = AGTACGCA$

$b = TATGC$

удаления и вставки -2

совпадения 2

мутации -1

$H(AGTACGCA, TATGC)$

$NW(AGTA, b)$

| | | T | A | T | G | C |
|---|----|----|----|----|----|-----|
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | 0 | -2 | -4 | -6 |
| G | -4 | -3 | -2 | -1 | 0 | -2 |
| T | -6 | -2 | -4 | 0 | -2 | -1 |
| A | -8 | -4 | 0 | -2 | -1 | -3 |

$NW(\text{rev}(CGCA), \text{rev}(b))$

| | | C | G | T | A | T |
|---|----|----|----|----|----|-----|
| | 0 | -2 | -4 | -6 | -8 | -10 |
| A | -2 | -1 | -3 | -5 | -4 | -6 |
| C | -4 | 0 | -2 | -4 | -6 | -5 |
| G | -6 | -2 | 2 | 0 | -2 | -4 |
| C | -8 | -4 | 0 | 1 | -1 | -3 |

ScoreL = $[-8, -4, 0, -2, -1, -3]$
 rev(ScoreR) = $[-3, -1, 1, 0, -4, -8]$
 Sum = $[-11, -5, 1, -2, -5, -11]$

Алгоритм Хиршберга. Пример.

$a = AGTACGCA$

$b = TATGC$

удаления и вставки — 2

совпадения 2

мутации — 1



Алгоритм Хиршберга. Пример.

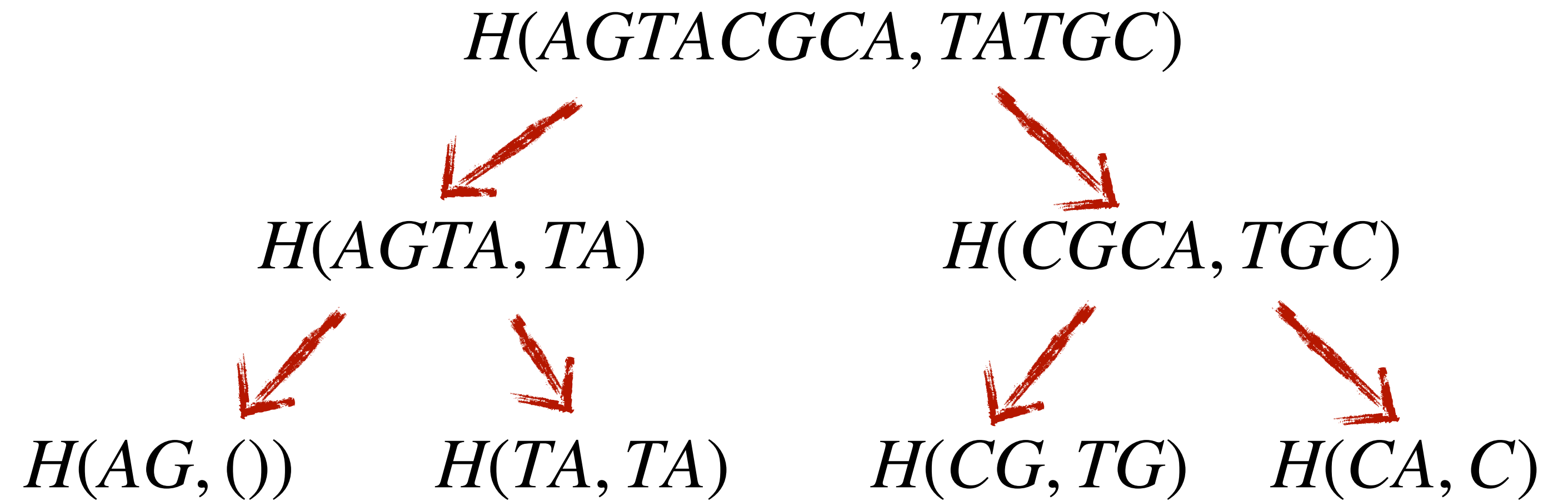
$a = AGTACGCA$

$b = TATGC$

удаления и вставки — 2

совпадения 2

мутации — 1



Алгоритм Хиршберга. Пример.

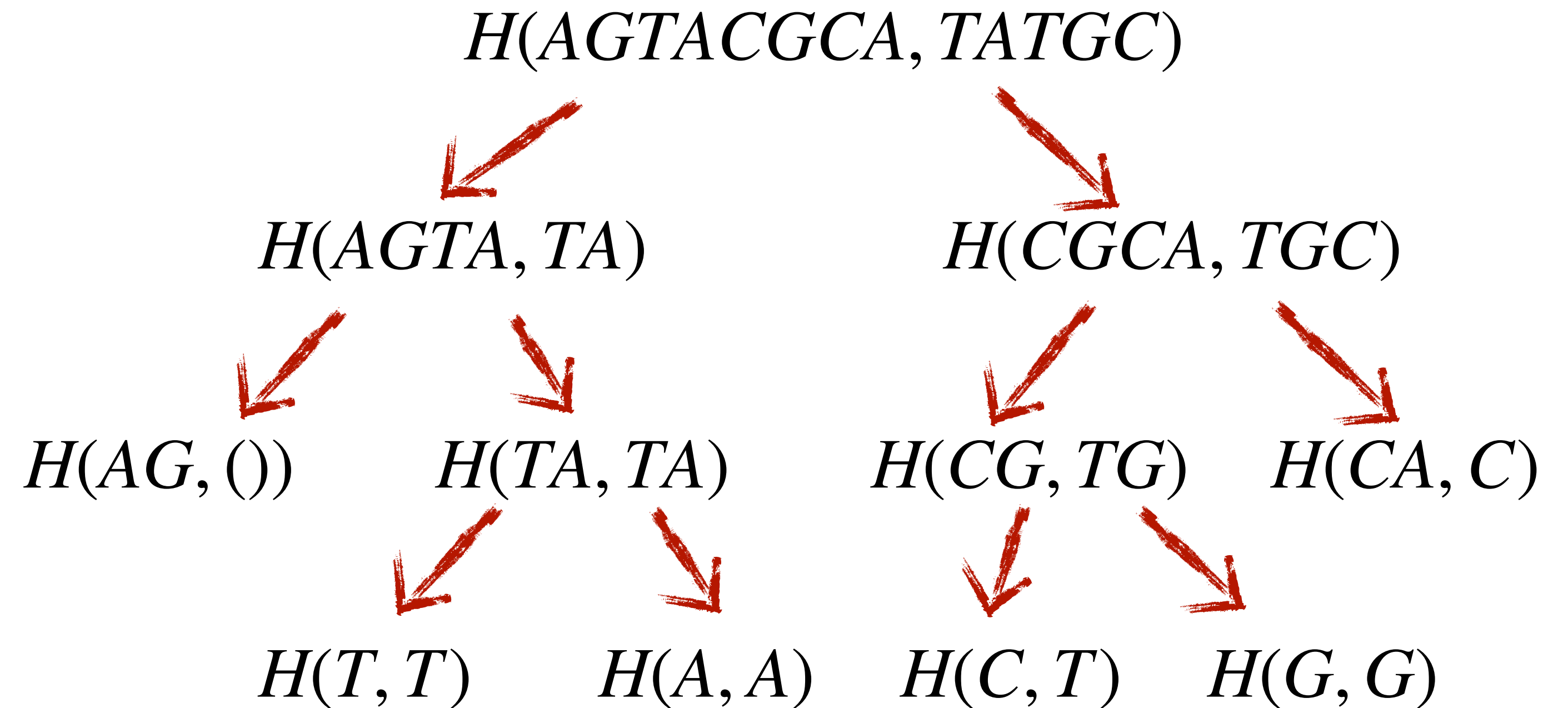
$a = AGTACGCA$

$b = TATGC$

удаления и вставки — 2

совпадения 2

мутации — 1



Алгоритм Хиршберга. Сложность.

- Сумма количества операций на каждом шаге

Алгоритм Хиршберга. Сложность.

- Сумма количества операций на каждом шаге

$$nm +$$

$$\frac{n}{2}(m - i) + \frac{n}{2}(i) +$$

$$\frac{n}{4}(m - i - j) + \frac{n}{4}(j) + \frac{n}{4}(i - k) + \frac{n}{4}(k) +$$

$$\dots =$$

$$= nm + \frac{n}{2}m + \frac{n}{4}m + \dots = nm \sum_{t=0}^{\log_2(n)} \frac{1}{2^t} \leq 2nm \Rightarrow O(nm)$$

Алгоритм Хиршберга. Сложность.

- Другой способ — решить рекуррентное соотношение (самостоятельно)

$$T(n, m) = T\left(\frac{n}{2}, m - j\right) + T\left(\frac{n}{2}, j\right) + O(nm), \text{ где } j \in (0, m)$$

Алгоритм Хиршберга. Оценка.

1. По памяти

$$O(n)$$

2. По времени

$$O(n^2)$$

3. Используется асимптотически меньше памяти, а скорость хуже только на константу!

Алгоритм на основе решения дифуров

Оказывается можно рассмотреть уравнения динамики как систему дифференциальных уравнений

Suzuki H., Kasahara M.
Introducing difference recurrence relations for faster semi-global alignment of long sequences
BMC bioinformatics 2018
<https://github.com/ocxtal/libgaba>

Wavefront Алгоритм

Ограничимся такими условиями

Штрафы g и e - за открытие, e - за продолжение

Совпадения - 0

Мутации - x

Тогда можно найти выравнивание за $O(n * s)$ где s - это вес оптимального выравнивания!

Wavefront Алгоритм

Приходите на JC

Marco-Sola, Santiago, et al.

Fast gap-affine pairwise alignment using the wavefront algorithm
Bioinformatics 2020

<https://github.com/smarco/WFA>

Резюмируем

- Если нам важно искать самое лучшее совпадение подпоследовательностей, то локально выравниваем.
- Можно использовать линию памяти почти не теряя в скорости
- Не забываем по JS!