Алгоритм поиска диссонансов временного ряда

Определения

Временной ряд (time series) T представляет собой хронологически упорядоченную последовательность вещественных значений $t_1, t_2, ..., t_N$, ассоциированных с отметками времени, где N – длина последовательности.

Подпоследовательность (subsequence) T_{im} временного ряда T представляет собой непрерывное подмножество T из m элементов, начиная с позиции i, т.е. $T_{im} = t_i, t_{i+1}, ..., t_{i+m-1}$, где $1 \leq i \leq N$ и $i + m \leq N$.

Расстояние (distance) между подпоследовательностями C и M представляет собой функцию, которая в качестве аргументов принимает C и M, и возвращает неотрицательное число R. Для подпоследовательностей функция расстояния является симметричной, т.е. $Dist(C, M) = Dist(M, C)$.

Пусть имеется временной ряд T, подпоследовательность C длины n, начинающаяся с позиции p, и подпоследовательность M длины n, начинающаяся с позиции q. Подпоследовательности C и M называются *несамопересекающимися (non-self match)*, если $\mid p - q \mid \geq n$.

Определения

Подпоследовательность D длины n, начинающаяся с позиции l называется *диссонансом* временного ряда T, если D находится на наибольшем расстоянии от ближайшей несамопересекающейся с D подпоследовательности, чем любые другие подпоследовательсти временного ряда, т.е. $\forall C \in T$, несамопересекающихся с D M_D и с C $M_C : min(Dist(D, M_D)) > min(Dist(C, M_C))$.

Подпоследовательность D длины n, начинающаяся с позиции p называется *K-м диссонансом временного ряда*, если D имеет K-е по размеру расстояние от ближайшей несамопересекающейся подпоследовательности, при этом не имея пересекающихся частей с i-ми диссонансами, начинающимися на позициях p_i, для всех $1 \leq i \leq K$, т.е. $\mid p - p_i \mid > n$.

Евклидово расстояние между двумя временными рядами Q и C длины n вычисляется по формуле 1:

$$Dist(Q, C) = \sqrt{\sum_{i=1}^{n} (q_i - c_i)^2} \tag{1}$$

Алгоритм:

1. Подготовка (выбор эвристики) – подбор порядка подачи подпоследовательностей, при котором возможно быстро отбрасывать неподходящие подпоследовательности.
2. Поиск диссонансов – перебор упорядоченных подпоследовательностей временного ряда с поиском наибольшего расстояния до ближайшего соседа

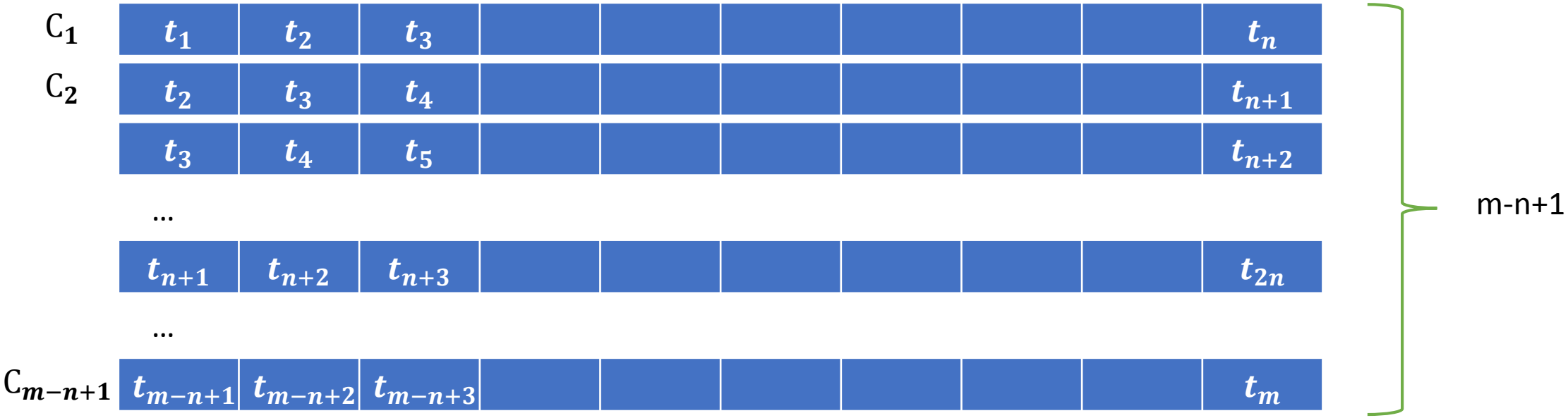
Входные данные

- T – временной ряд ($t_i, 1 \leq i \leq m$)
- m – длина ряда
- n – длина подпоследовательности
- C – множество подпоследовательностей
- w – длина слова (в SAX аппроксимации подпоследовательностей), $1 \leq w \leq n, n \; mod \; w = 0$
- Lookup table (LT) – для точек разделения в SAX
- A – мощность алфавита для SAX представления подпоследовательностей

Хранение подпоследовательностей временного ряда

Временной ряд T:

Подпоследовательности C:

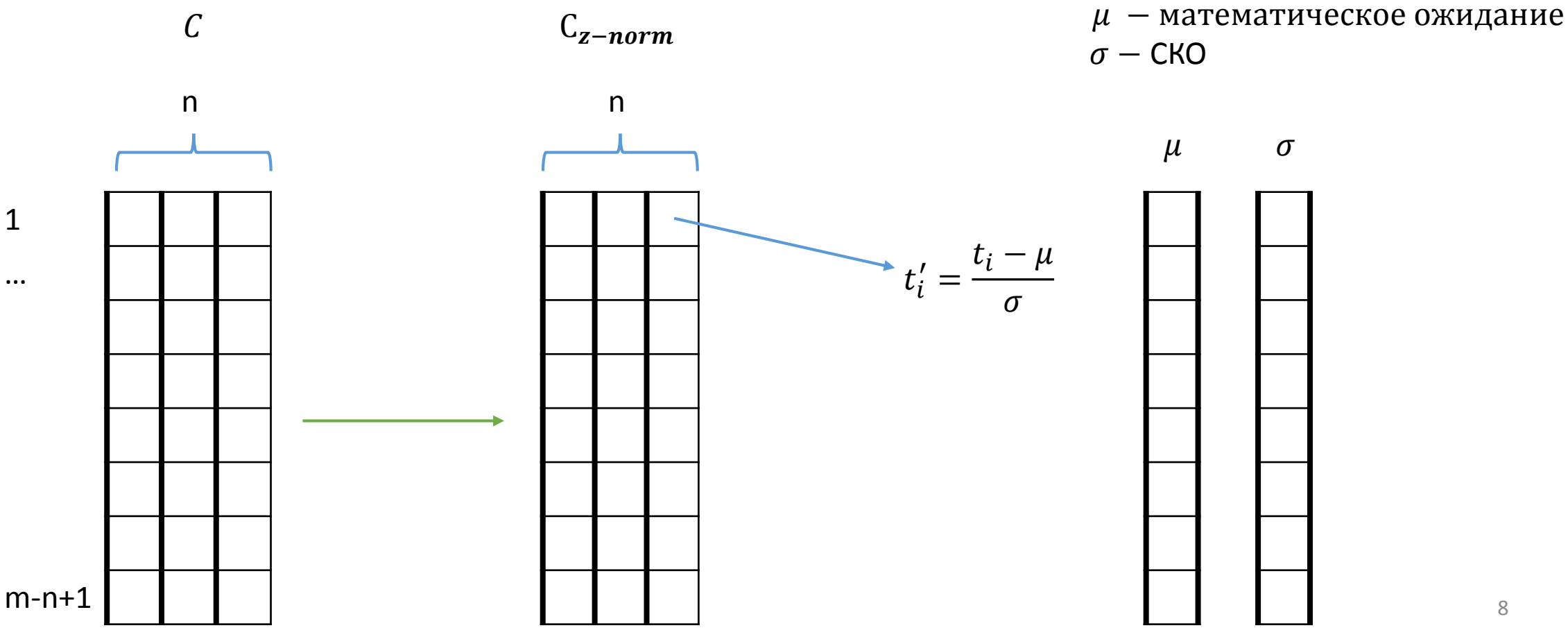


Алгоритм:

1. Подготовка (выбор эвристики) – подбор порядка подачи подпоследовательностей, при котором возможно быстро отбрасывать неподходящие подпоследовательности.
    1. Z-нормализация подпоследовательностей C_i временного ряда
    2. Кусочная аппроксимация (PAA-представление)
    3. Кодирование с помощью lookup table
    4. Подсчет частот, нахождение мин. значения
2. Поиск диссонансов – перебор упорядоченных подпоследовательностей временного ряда с поиском наибольшего расстояния до ближайшего соседа

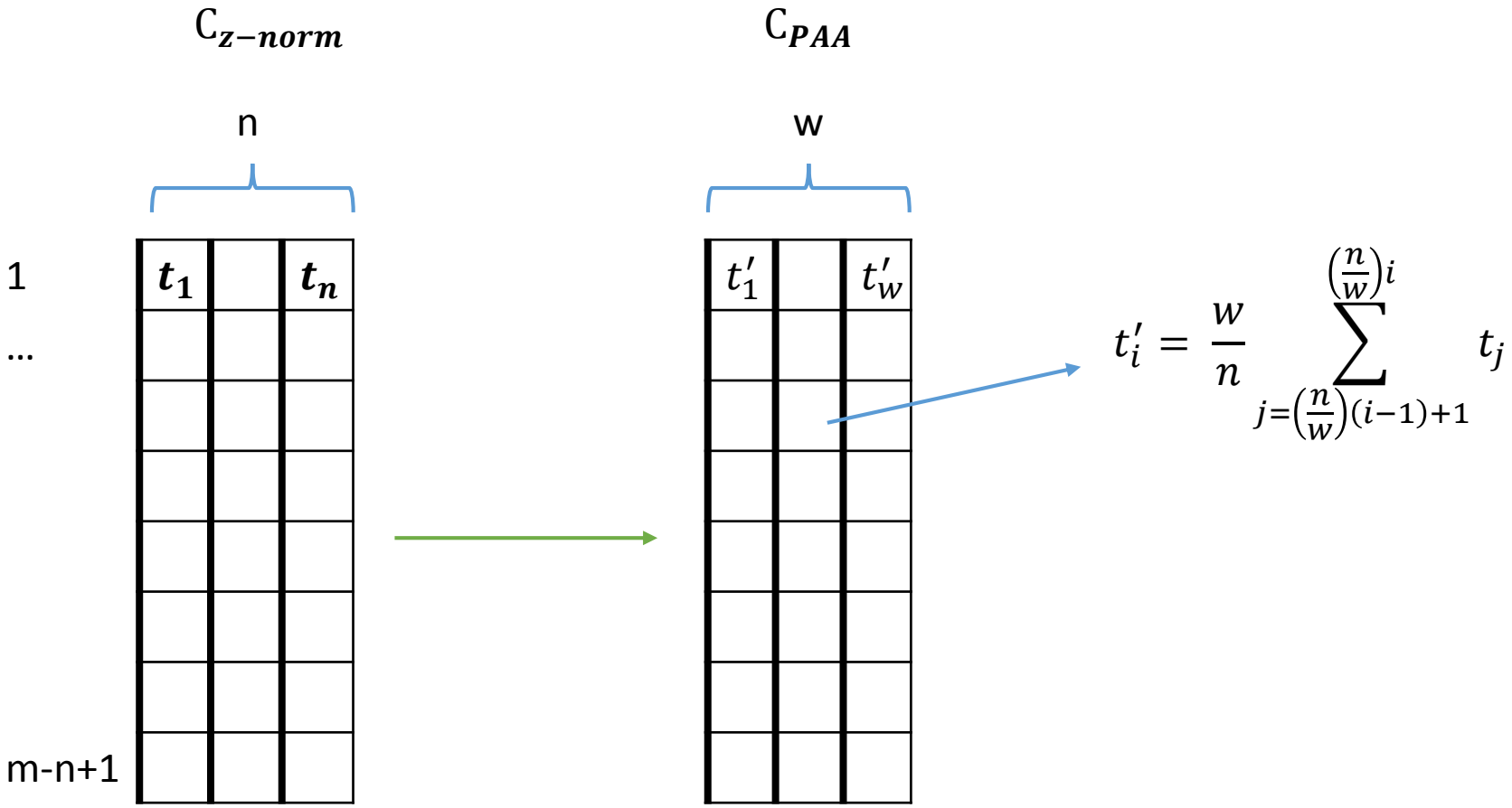
1. Подбор эвристики

1.1 z-нормализация подпоследовательностей C_i временного ряда



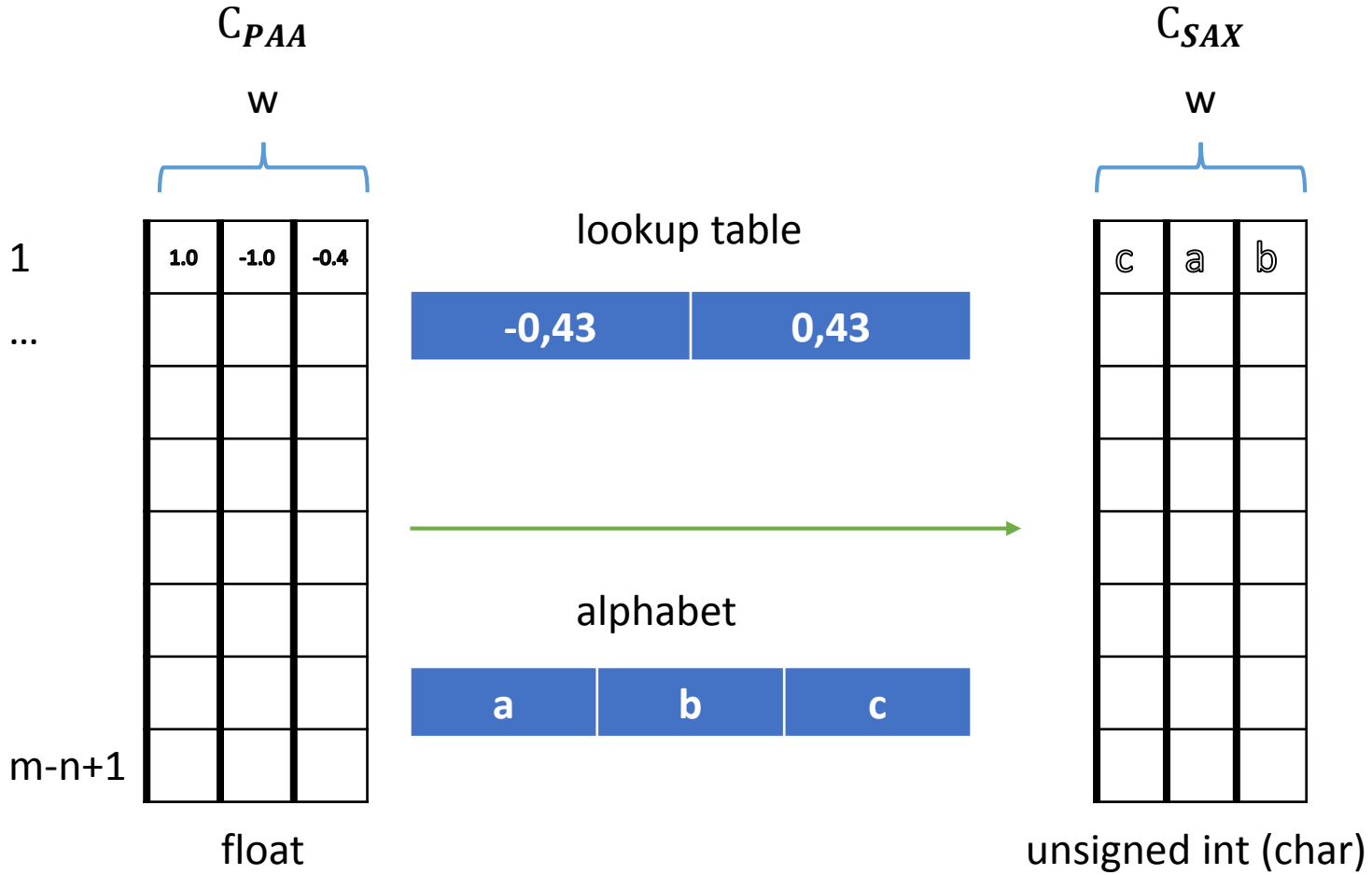
1. Подбор эвристики

1.2 Аппроксимация с помощью кусочной агрегации (PAA)



1. Подбор эвристики

1.3 Кодирование с помощью lookup table (аппроксимация с помощью символьной агрегации)

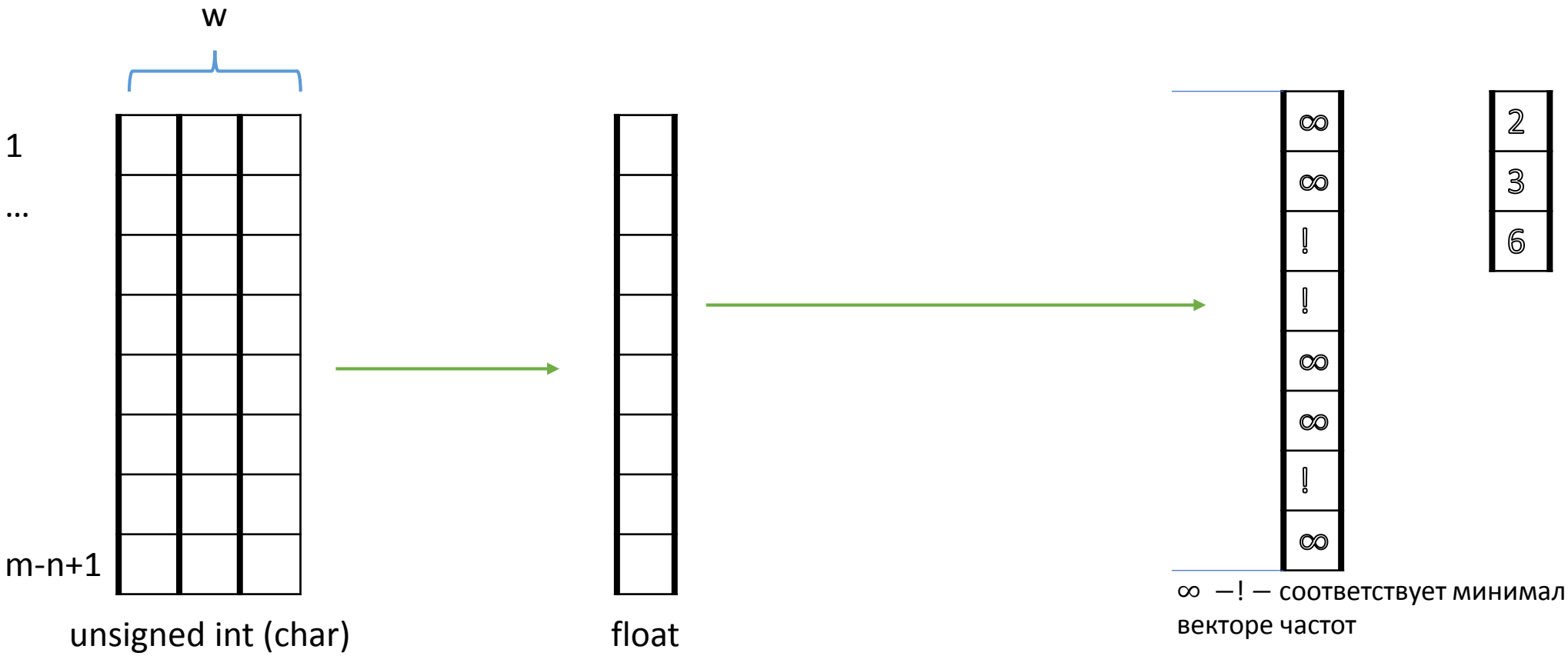


1. Подбор эвристики

1.4 Подсчет частот, нахождение мин. значения

C_{SAX}

Frequences – частота вхождения каждого из слов в SAX представлении

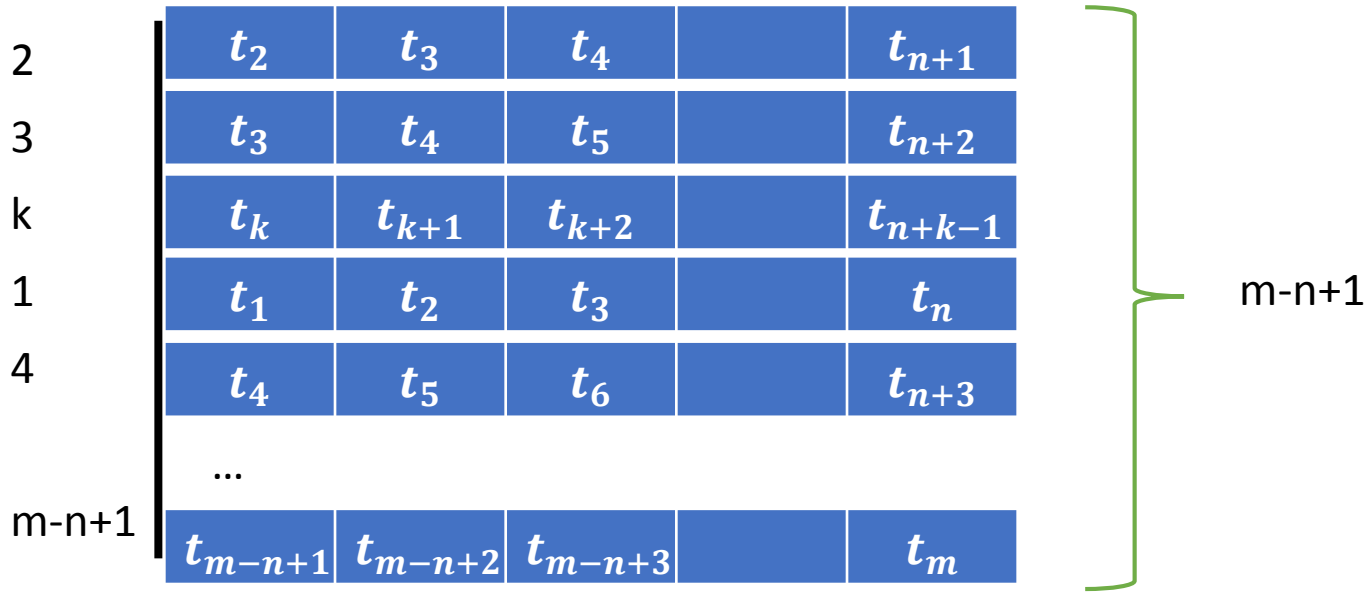


∞ —! — соответствует минимальному значению в векторе частот

Упорядочивание множества подпоследовательностей

Index minVal

2
3
k



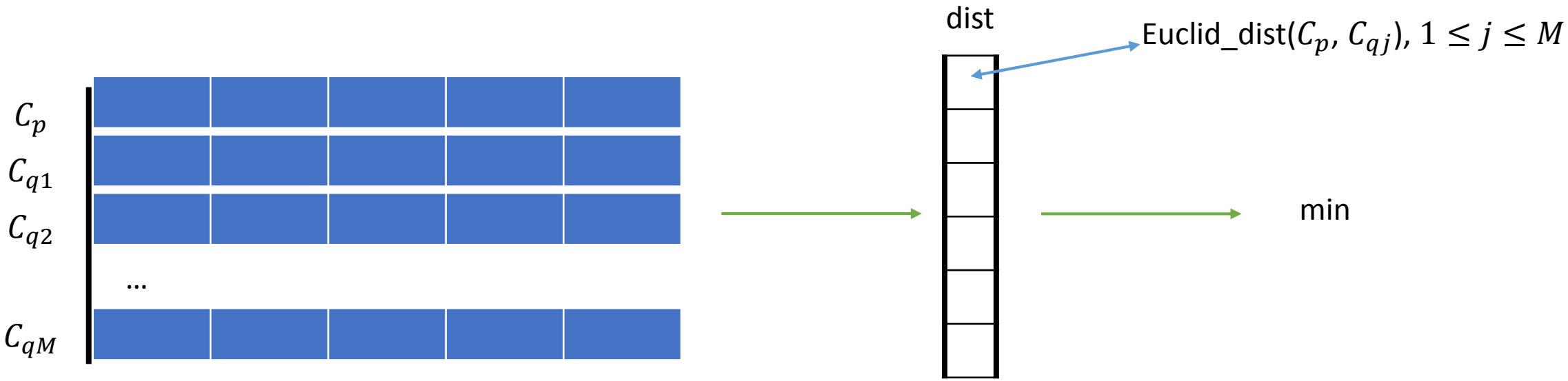
Алгоритм:

1. Подготовка (выбор эвристики) – подбор порядка подачи подпоследовательностей, при котором возможно быстро отбрасывать неподходящие подпоследовательности.
2. Поиск диссонансов – перебор упорядоченных подпоследовательностей временного ряда с поиском наибольшего расстояния до ближайшего соседа
    1. Нахождение оптимальной best_so_far_dist (для подпоследовательностей - предположительных диссонансов)
    2. Нахождение расстояния до ближайшего соседа для оставшихся подпоследовательностей

- 2а – на основе последовательного алгоритма HOTSAX Кеога. На основе полученной эвристики выбирается оптимальный порядок перебора подпоследовательностей при поиске расстояния до ближайшего соседа. При этом находить расстояние до ближайшего соседа придется только для нескольких первых подпоследовательностей. Для оставшихся будет срабатывать условие «раннего выхода» из цикла.
- 2б – составить матрицу расстояний для всех подпоследовательностей (для self-match будут фиктивные расстояния). Затем найти максимум из минимумов расстояний.

2. Поиск диссонансов

2.1 Нахождение расстояния до ближайшего соседа для всех подпоследовательностей с обновлением best_so_far_dist

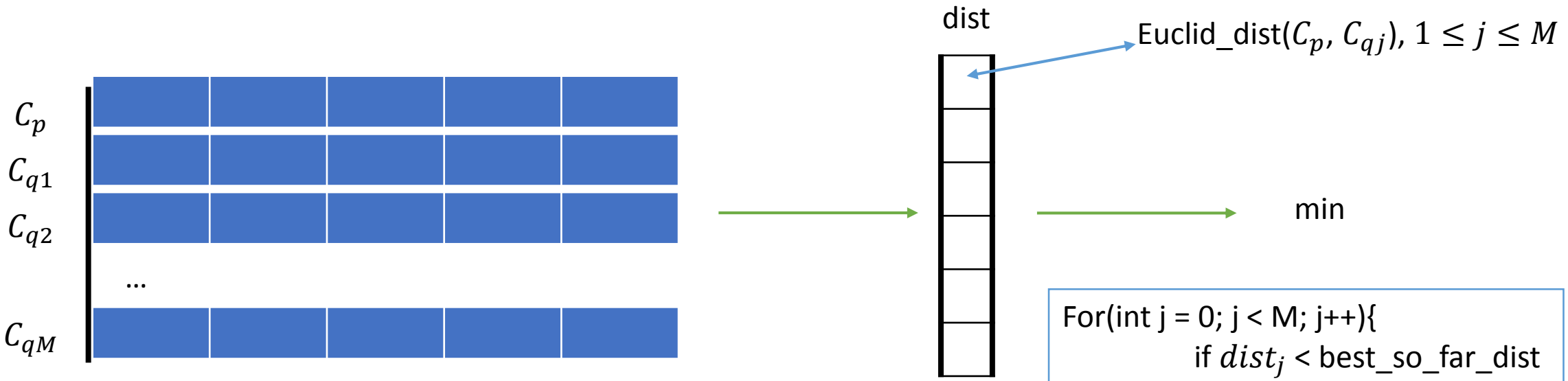


C_p – подпоследовательность, для которой нужно найти расстояние до ближайшего соседа

$\{C_{qi}\}$ – множество $not - selfmatch$ с C_p подпоследовательностей

2. Поиск диссонансов

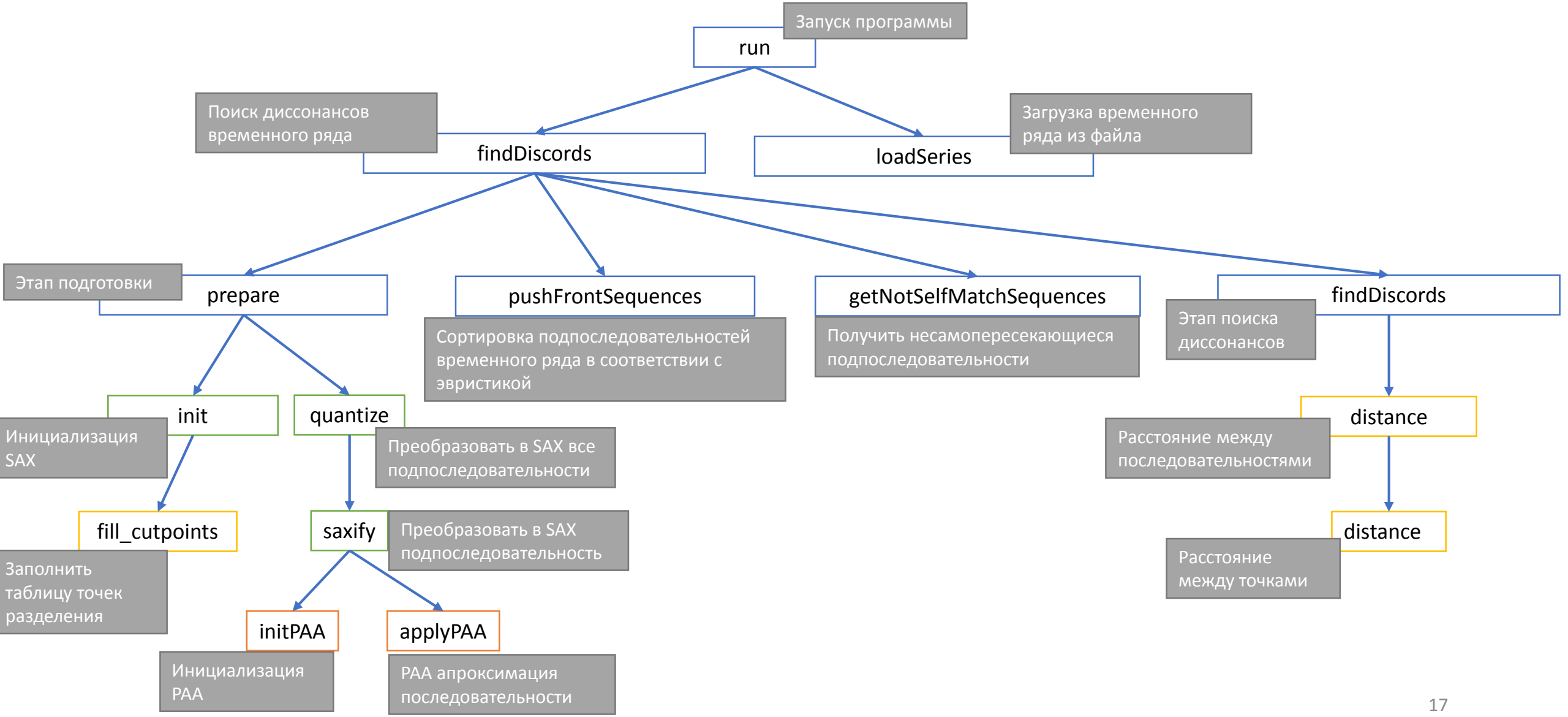
2.2 Нахождение расстояния до ближайшего соседа для оставшихся подпоследовательностей



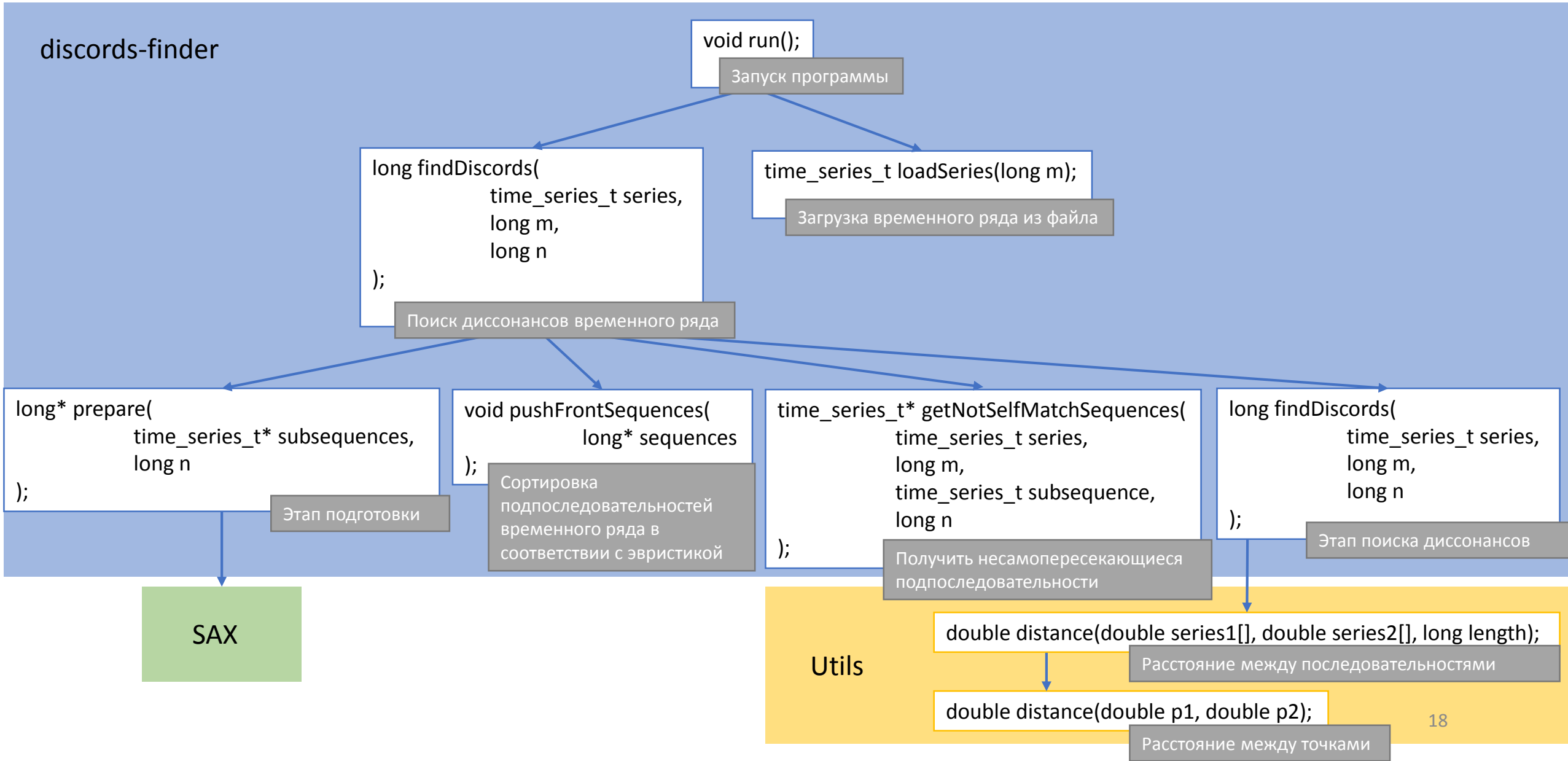

```
For(int j = 0; j < M; j++){
        if dist_j < best_so_far_dist
                -> не подходит
}
If(min > best_so_far_dist){
        best_so_far_dist = min;
        best_so_far_pos = p;
}
```

Output: диссонансом является подпоследовательность начинающаяся с best_so_far_pos позиции в исходном временном ряде.

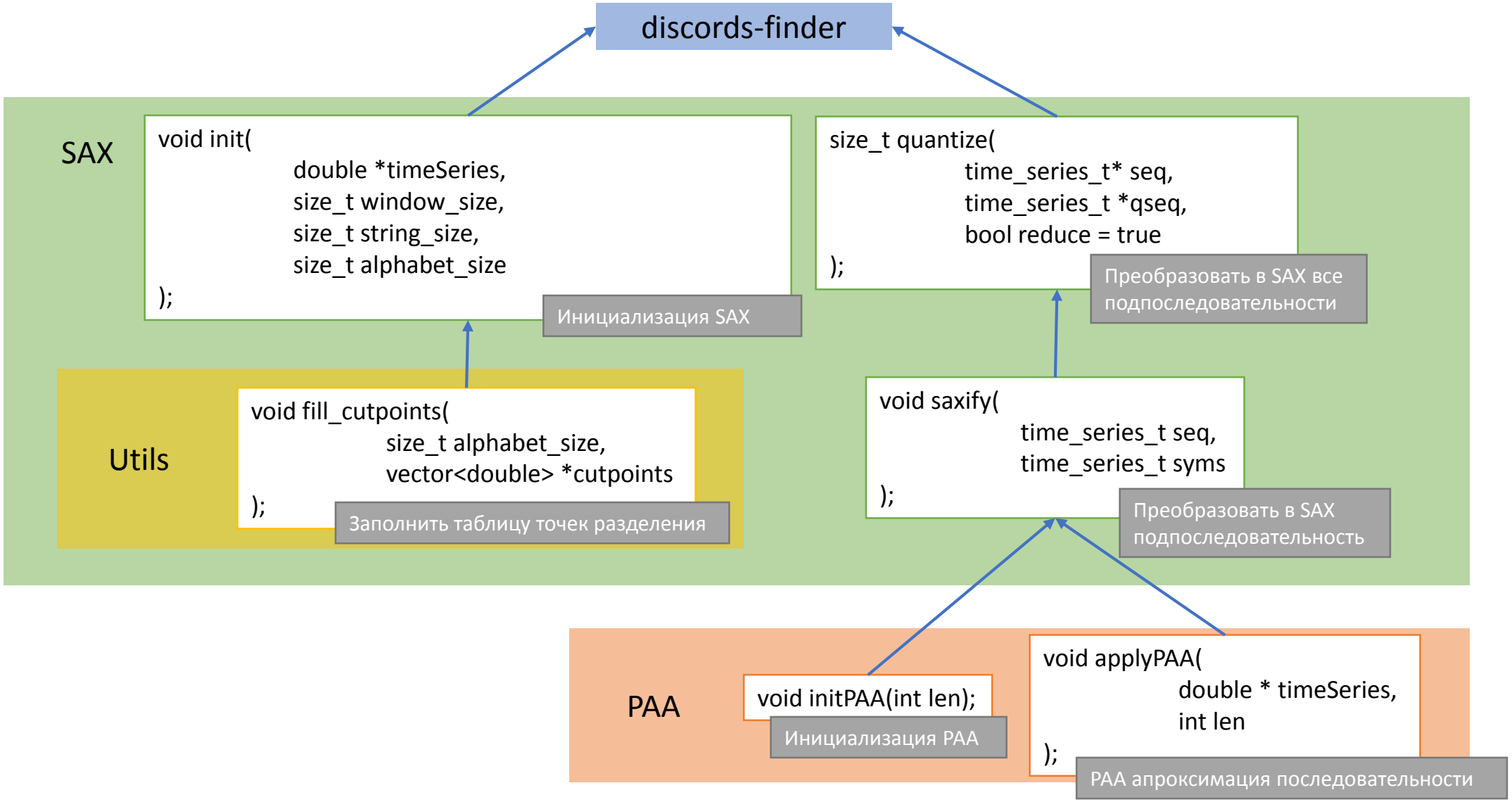
Модульная структура



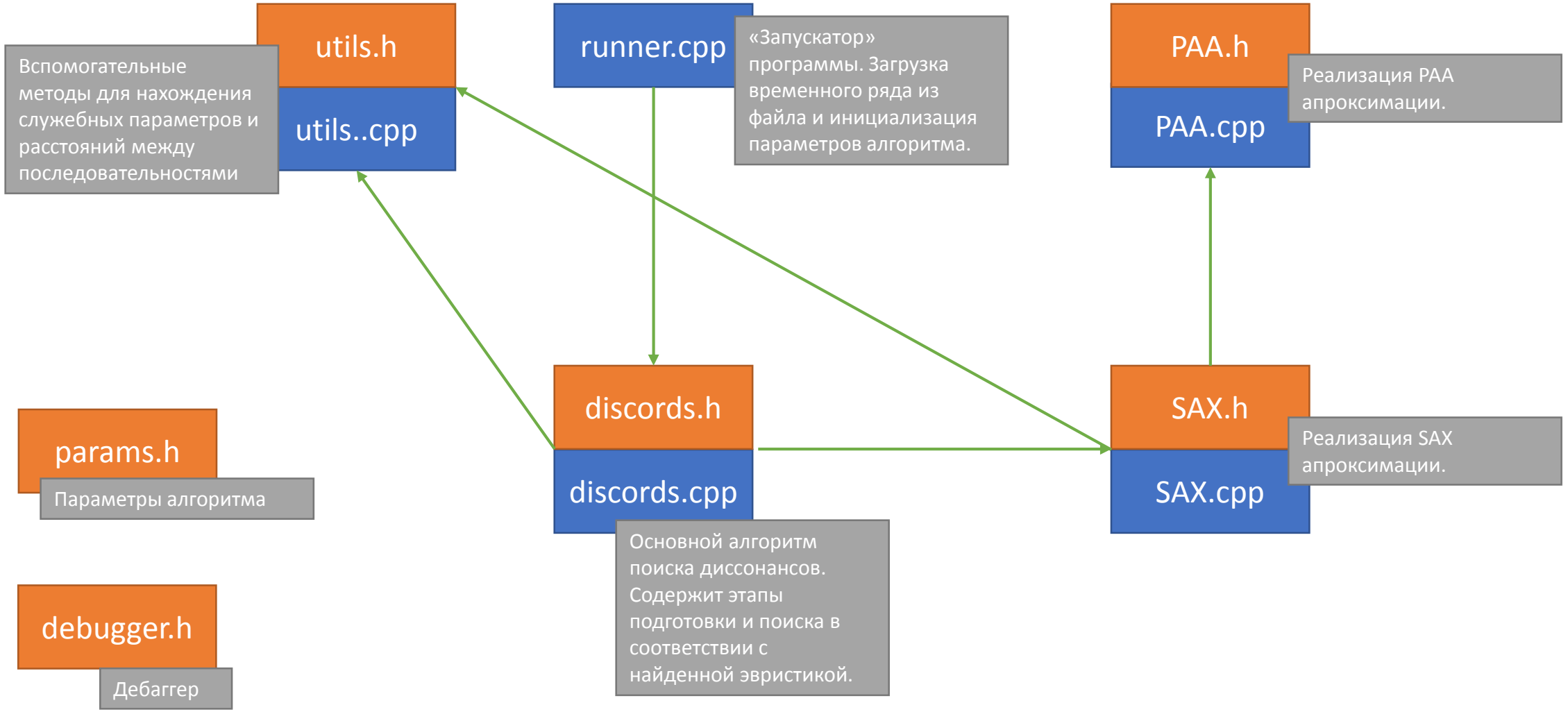
Модульная структура



Модульная структура



Файловая структура



Файловая структура