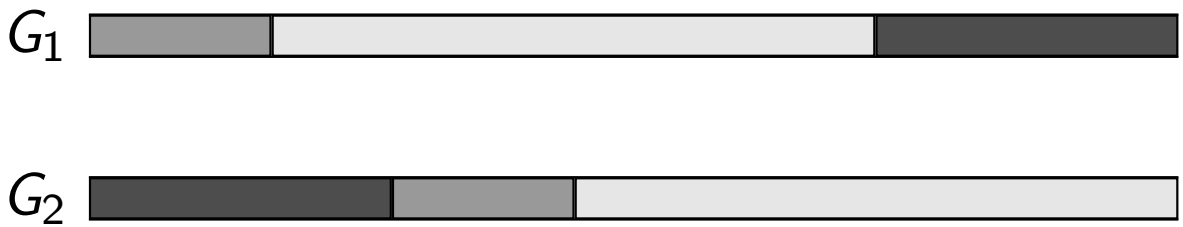
The maximal matches model for sequence comparison

- The edit distance is a measure which is highly dependent on the relative positions of matches in the compared sequence sequences.
- There are situations where this property is not desired.
- Suppose one wants to consider genomes as similar which differ only by an exchange of large substrings.
- This occurs, for instance, if a bacterial genome that has evolved from an ancestral genome by transposition of large sections of DNA, see the following illustration with two genomes G_1 and G_2, such that blocks of the same grey-scale are considered to be highly similar:

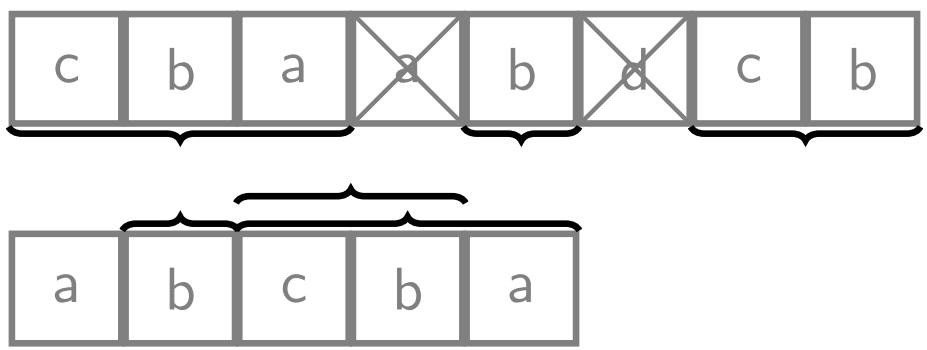


The maximal matches model for sequence comparison

- In such a case, the edit distance would be very large so that it is likely, that the overall similarities are not detected (because one would filter for pairs of genomes with small distances).
- Furthermore, the computation of the edit distance is very costly in terms of the running time.
- So when e.g. comparing thousands of genomes with each other (which is a typical task today), it would take a long time to compute all pairwise edit distances.
- For these reasons, alternatives to the edit distance model were developed.
- Here we consider three such alternative comparison models that do not care about the order of matches and can be computed in linear time: The maximal matches model, the q-gram model and the *minHash* model.

The maximal matches model for sequence comparison

- The idea of this model, first described in [Ehrenfeucht and Haussler, 1988] is to measure the distance between strings in terms of common substrings.
- Strings are considered similar if they have long common substrings, independent of where they occur.
- Technically, one counts the minimum number of occurrences of characters in one sequence such that if these characters are "crossed out", the remaining substrings are all substrings of the other sequence, as shown in the following illustration:



- The key to the model is the notion of partition.
- Recall that u and v are strings of length m and n, respectively.

The maximal matches model for sequence comparison

Definition 1

- A *partition* of v with respect to u is a sequence $(w_1, c_1, \ldots, w_r, c_r, w_{r+1})$ of substrings $w_1, \ldots, w_r, w_{r+1}$ of u and characters $c_1, \ldots, c_r$ such that $v = w_1 c_1 \ldots w_r c_r w_{r+1}$.
- Let $\Psi = (w_1, c_1, \ldots, w_r, c_r, w_{r+1})$ be a partition of v with respect to u (Ψ is pronounced as Psi) .
- $w_1, \ldots, w_r, w_{r+1}$ are the *submatches* in Ψ.
- $c_1, \ldots, c_r$ are the *marked characters* in Ψ.
- The size of Ψ, denoted by $|\Psi|$, is r.
- $mmdist(v, u)$ is the size of any minimal partition of v with respect to u.
- We call $mmdist(v, u)$ *maximal matches distance* of v and u.

Figure 1: Illustration of a partition of sequence v with respect to u. The blocks of different grey-scales are the submatches separated by marked characters. These blocks match substrings of the same grey-scale in sequence u. Additionally, the correspondence of submatches in v to substrings in u is shown by dotted lines connecting the left- and rightmost positions of involved strings. The substrings matched in u can overlap, as is the case for the first two substrings. The size of the partition is 3.

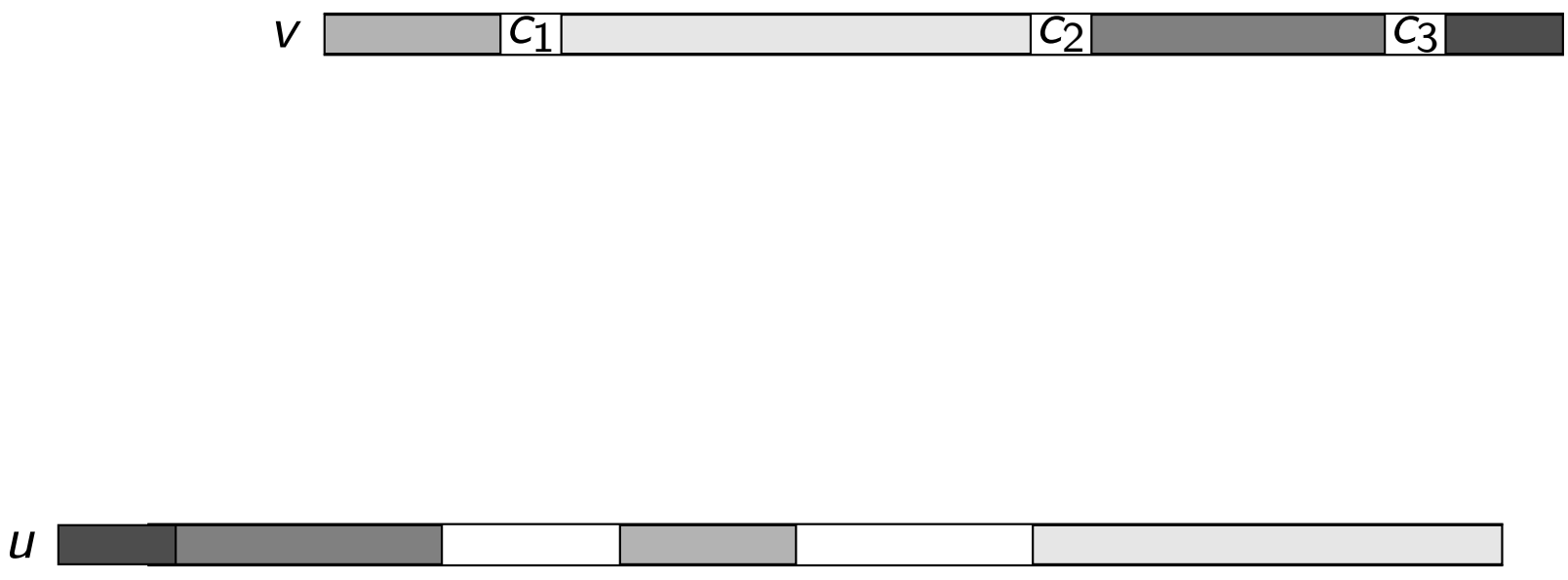


Figure 1: Illustration of a partition of sequence v with respect to u. The blocks of different grey-scales are the submatches separated by marked characters. These blocks match substrings of the same grey-scale in sequence u. Additionally, the correspondence of submatches in v to substrings in u is shown by dotted lines connecting the left- and rightmost positions of involved strings. The substrings matched in u can overlap, as is the case for the first two substrings. The size of the partition is 3.

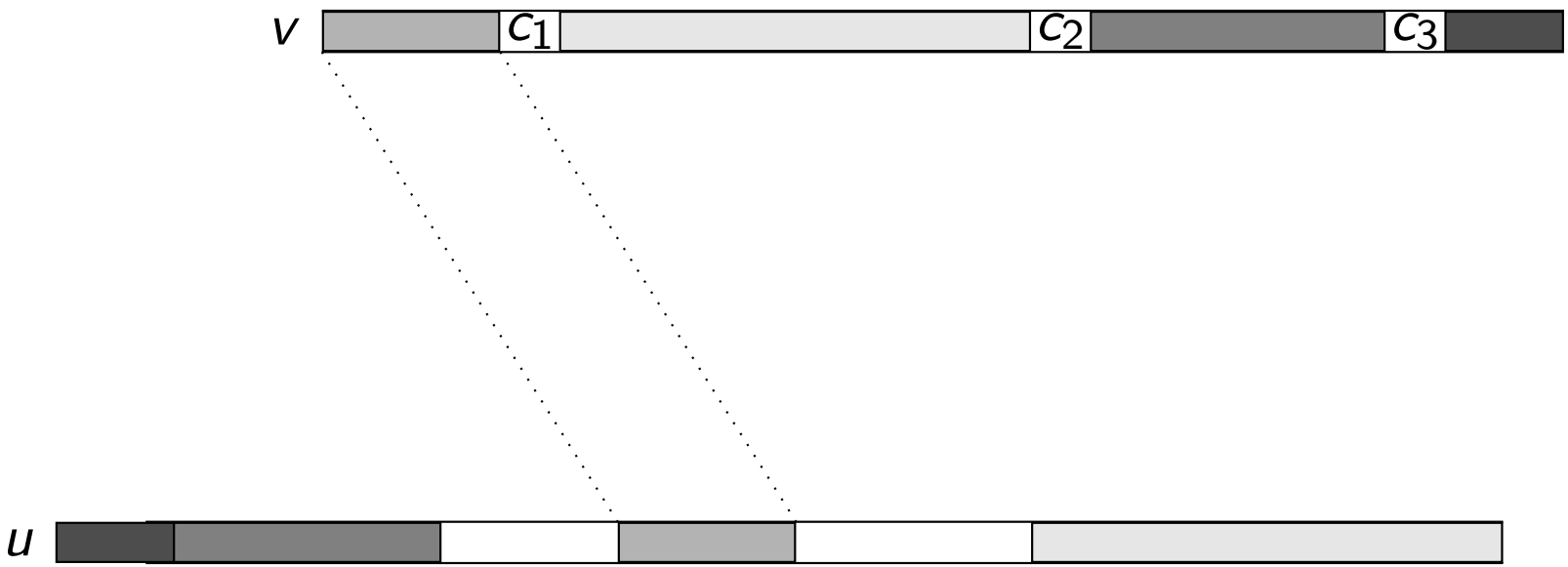


Figure 1: Illustration of a partition of sequence v with respect to u. The blocks of different grey-scales are the submatches separated by marked characters. These blocks match substrings of the same grey-scale in sequence u. Additionally, the correspondence of submatches in v to substrings in u is shown by dotted lines connecting the left- and rightmost positions of involved strings. The substrings matched in u can overlap, as is the case for the first two substrings. The size of the partition is 3.

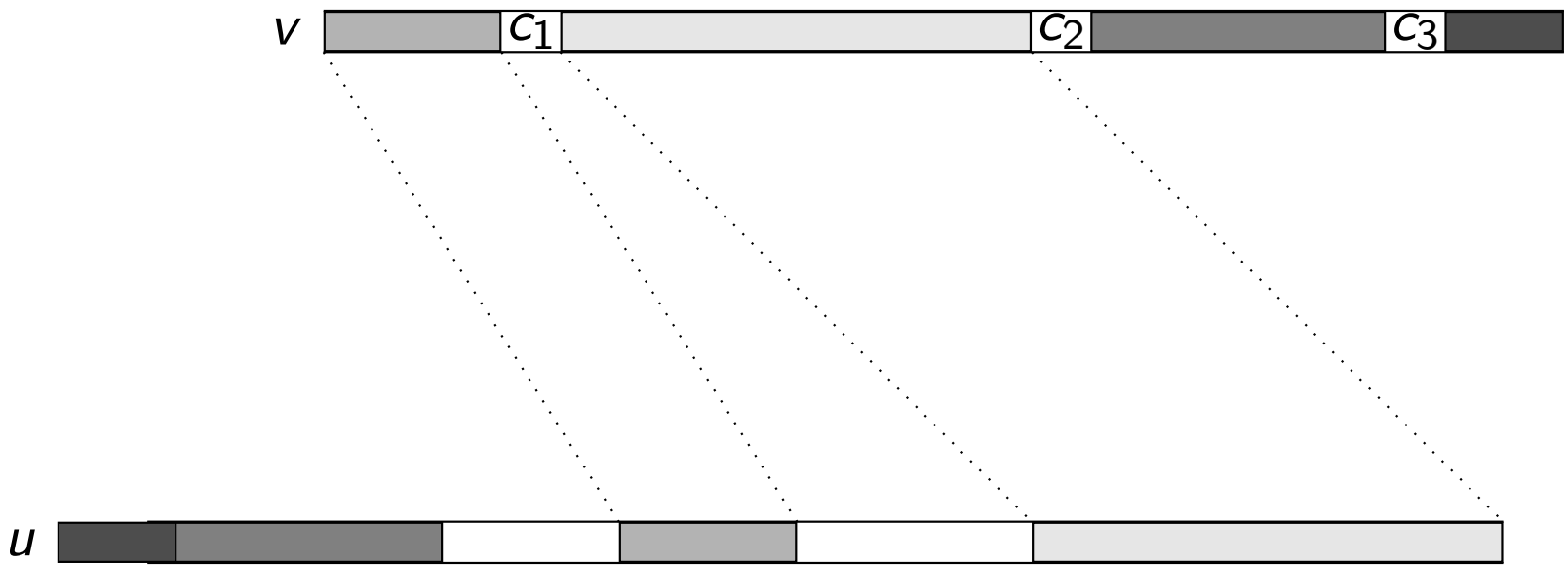


Figure 1: Illustration of a partition of sequence v with respect to u. The blocks of different grey-scales are the submatches separated by marked characters. These blocks match substrings of the same grey-scale in sequence u. Additionally, the correspondence of submatches in v to substrings in u is shown by dotted lines connecting the left- and rightmost positions of involved strings. The substrings matched in u can overlap, as is the case for the first two substrings. The size of the partition is 3.

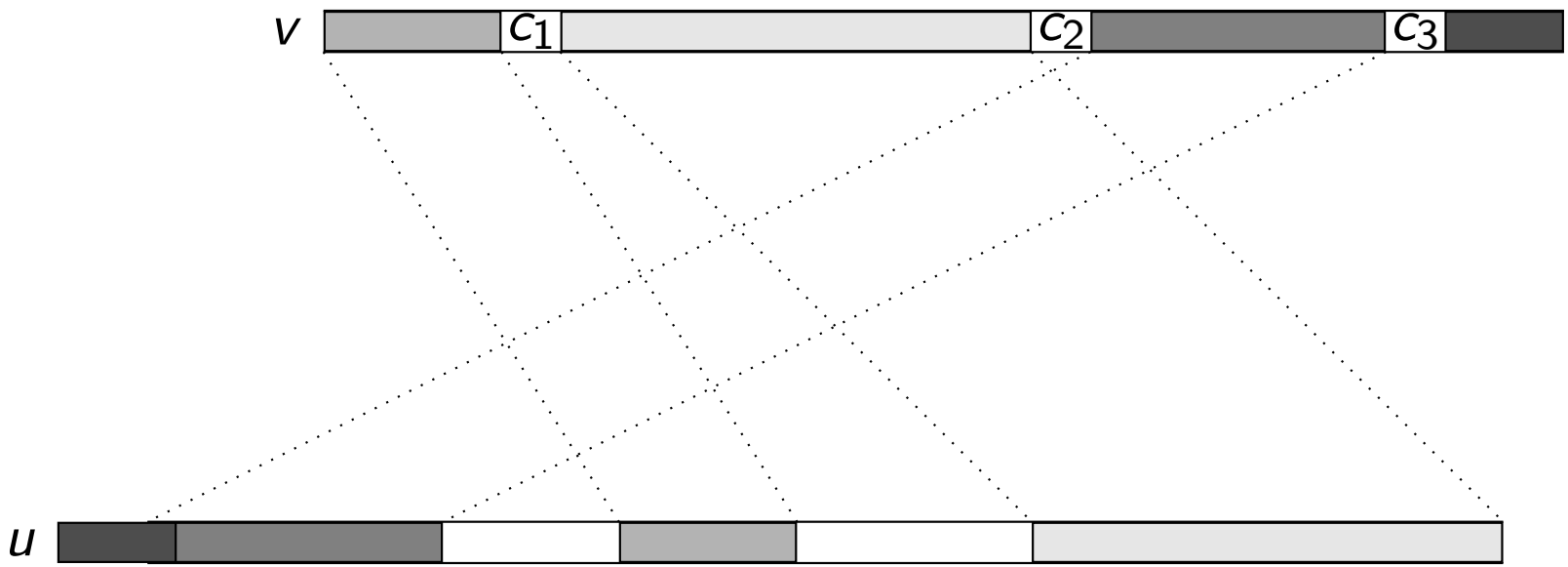
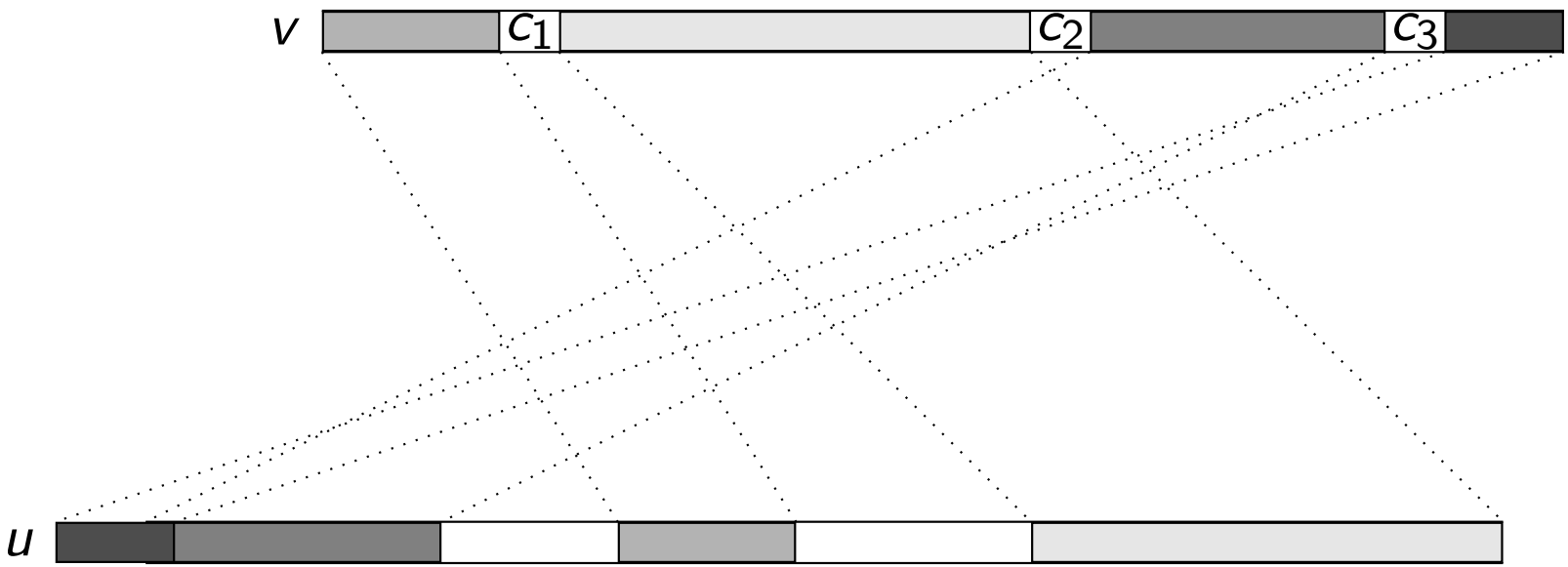


Figure 1: Illustration of a partition of sequence v with respect to u. The blocks of different grey-scales are the submatches separated by marked characters. These blocks match substrings of the same grey-scale in sequence u. Additionally, the correspondence of submatches in v to substrings in u is shown by dotted lines connecting the left- and rightmost positions of involved strings. The substrings matched in u can overlap, as is the case for the first two substrings. The size of the partition is 3.



Example 1

- Let $v = \texttt{cbaabdcb}$ and $u = \texttt{abcba}$.
- $\Psi_1 = (\texttt{cba}, \texttt{a}, \texttt{b}, \texttt{d}, \texttt{cb})$ is a partition of v with respect to u, since cba, b, and cb are substrings of u.
- $\Psi_2 = (\texttt{cb}, \texttt{a}, \texttt{ab}, \texttt{d}, \texttt{cb})$ is a partition of v with respect to u, since cb and ab are substrings of u.
- Both partitions are of size 2.
- Suppose there is a partition of v with respect to u of size less than 2.
- This would be a partition with zero or one marked character, requiring that there is a submatch of size at least $\left\lceil \frac{|v|-1}{2} \right\rceil = \left\lceil \frac{8-1}{2} \right\rceil = 4$.
- But as the longest common substring of v and u is 3, such a partition does not exist.
- So Ψ_1 and Ψ_2 are of minimal size.
- Hence, $mmdist(v, u) = 2$.

The maximal matches model for sequence comparison

Definition 2

- Let $\Psi = (w_1, c_1, \ldots, w_r, c_r, w_{r+1})$ be a partition of v with respect to u.
- If for all h, $1 \leq h \leq r$, $w_h c_h$ is not a substring of u, then Ψ is the *left-to-right partition* of v with respect to u.
- If for all h, $1 \leq h \leq r$, $c_h w_{h+1}$ is not a substring of u, then Ψ is the *right-to-left partition* of v with respect to u.
- The left-to-right partition of v with respect to u is denoted by $\Psi_{lr}(v, u)$.
- The right-to-left partition of v with respect to u is denoted by $\Psi_{rl}(v, u)$.

Example 2

For the strings $v = \texttt{cbaabdcb}$ and $u = \texttt{abcba}$ of Example 1 we have:

- $\Psi_{lr}(v, u) = \Psi_1 = (\texttt{cba}, \texttt{a}, \texttt{b}, \texttt{d}, \texttt{cb})$ as $\texttt{cbaa}$ and $\texttt{bd}$ are not substrings of u.
- $\Psi_{rl}(v, u) = \Psi_2 = (\texttt{cb}, \texttt{a}, \texttt{ab}, \texttt{d}, \texttt{cb})$ as $\texttt{dcb}$ and $\texttt{aab}$ are not substrings of u.

Example 3

For the strings $v' = \texttt{abcba}$ and $u' = \texttt{cbaabdcb}$. We have:

- $\Psi_{lr}(v', u') = (\texttt{ab}, \texttt{c}, \texttt{ba})$ as $\texttt{abc}$ is not a substring of u'.
- $\Psi_{rl}(v', u') = (\texttt{a}, \texttt{b}, \texttt{cba})$ as $\texttt{bcba}$ is not a substring of u.

The maximal matches model for sequence comparison

- One can show that $\Psi_{lr}(v, u)$ and $\Psi_{rl}(v, u)$ are of minimal size, see [Ehrenfeucht and Haussler, 1988].
- Hence, we can conclude $|\Psi_{lr}(v, u)| = mmdist(v, u) = |\Psi_{rl}(v, u)|$.
- This property leads to a simple algorithm for calculating the maximal matches distance.
- The partition $\Psi_{lr}(v, u)$ can be computed by scanning the characters of v from left to right, until a prefix wc of v is found such that w is a substring of u, but wc is not.
- w is the first submatch and c is the first marked character in $\Psi_{lr}(v, u)$.
- The remaining submatches and marked characters are obtained by repeating the process on the remaining suffix of v, until all of the characters of v have been scanned.
- Algorithm 1 gives pseudocode for the computation of the size of the left-to-right partition.

Algorithm 1 (Computation of left-to-right partition)

Input: sequences $u = u[1 \ldots m]$ and $v = v[1 \ldots n]$
Output: size of left-to-right-partition

```
1:  function lrpartitionsize(u,v)
2:      s ← ε                                   ▷ s: substring of u
3:      marked ← 0                              ▷ number of marked chars
4:      for idx = 1 to n do
5:          c ← v[idx]
6:          if sc is a substring of u then
7:              s ← sc
8:          else
9:              marked ← marked + 1             ▷ marked char. c at pos idx
10:             s ← ε
11:         end if
12:     end for
13:     return marked
14: end function
```

The maximal matches model for sequence comparison

- Using the suffix tree of u, denoted by $ST(u)$ (see course "Genome Informatics", next Summersemester), the longest prefix w of v that is a substring of u, can be computed in $O(|\mathcal{A}| \cdot |w|)$ time.
- This gives an algorithm to calculate $mmdist(v, u)$ in $O(|\mathcal{A}| \cdot (m + n))$ time and $O(m)$ space.
- $\Psi_{rl}(v, u)$ can be computed in a similar way by scanning v from right to left.
- However, one has to be careful since the reversed scanning direction means to compute the longest prefix of v^{-1} that occurs as substring of u^{-1}.
- This can, of course, be accomplished by using $ST(u^{-1})$ instead of $ST(u)$.

Example 4

Let us reconsider the strings $v = \texttt{cbaabdcb}$ and $u = \texttt{abcba}$ from Example 2. We have seen that

- $\Psi_{lr}(v, u) = (\texttt{cba}, \texttt{a}, \texttt{b}, \texttt{d}, \texttt{cb})$ which implies $\mathit{mmdist}(v, u) = 2$ and
- $\Psi_{lr}(u, v) = (\texttt{ab}, \texttt{c}, \texttt{ba})$ which implies $\mathit{mmdist}(u, v) = 1$

As $\mathit{mmdist}(v, u) = 2 \neq 1 = \mathit{mmdist}(u, v) = 1$, *mmdist* is not symmetric. Hence, *mmdist* is not a metric on $\mathcal{A}^*$.

However, one can obtain a metric as follows:

The maximal matches model for sequence comparison

Theorem 3

Let $mmm(u, v) = \log_2((mmdist(u, v) + 1) \cdot (mmdist(v, u) + 1))$. *mmm is a metric on* $\mathcal{A}^*$. $\square$

- From the above it is clear that $mmm(u, v)$ can be computed in $O(|\mathcal{A}| \cdot (m + n))$ time and $O(\max\{m, n\})$ space.
- Next we study the relation of the maximal matches distance and the unit edit distance.
- We first show an important relation of alignments and partitions.
- The idea is that an alignment can be turned into a partition in which consecutive matches form the submatches and characters replaced and inserted in v are the marked characters.

The maximal matches model for sequence comparison

Lemma 1

Let δ be the unit cost function. Consider an alignment A of v and u. Then there is an r, $0 \leq r \leq \delta(A)$, and a partition $(w_1, c_1, \ldots, w_r, c_r, w_{r+1})$ of v with respect to u such that w_1 is a prefix and w_{r+1} is a suffix of u.

proof: see lecture notes.

The following theorem shows that $mmdist(v, u)$ is a lower bound for the unit edit distance of v and u.

Theorem 4

Suppose δ is the unit cost function. Then $mmdist(v, u) \leq edist_\delta(v, u)$.

Proof.

- *Let A be an optimal alignment of v and u.*
- *Then by Lemma 1 there is a partition Ψ of v with respect to u of size $r \leq \delta(A)$.*
- *$mmdist(v, u) \leq |\Psi| = r \leq \delta(A) = edist_\delta(v, u)$.*

The maximal matches model for sequence comparison

- The relation between *mmdist* and $edist_\delta$ suggests to use *mmdist* as a filter in contexts where the unit edit distance is of interest only below some threshold k.
- For example, suppose we have a set S of sequences and want to find all pairs $(s, s') \in S \times S'$, $s \neq s'$ such that $edist_\delta(s, s') \leq k$.
- Instead of determining $edist_\delta(s, s')$ for each pair s, s' using $O(|s| \cdot |s'|)$ time, one first computes $mmdist(s, s')$ in $O(|s| + |s'|)$ time.
- If $mmdist(s, s') > k$, then by Theorem 4 we have $edist_\delta(s, s') > k$, so that the pair (s, s') does not have to be considered further.
- That is, *mmdist* serves as a filter for the expensive computation of the unit edit distance.
- If $mmdist(s, s') \leq k$, then we have to compute $edist_\delta(s, s')$.
- In fact, there are algorithms for the approximate sequence searching problem using filtering techniques based on maximal matches.


Ehrenfeucht, A. and Haussler, D. (1988).
A New Distance Metric on Strings Computable in Linear Time.
Discrete Applied Mathematics, 20:191–203.