Algorithms for Computational Biology

Zsuzsanna Lipták

Masters in Molecular and Medical Biotechnology
a.a. 2015/16, fall term

Strings and Sequences in Computer Science

Some formalism on strings (1)

- Σ a finite set called alphabet
- its elements are called characters or letters
- with $|\Sigma|$ we denote the size of the alphabet (number of different characters)
- a string over Σ is a finite sequence of characters from Σ
- we write strings as $s = s_1 s_2 \dots s_n$, where the s_i (for $i = 1, \dots, n$) are characters from Σ **N.B.**: We number strings from 1, not from 0
- $|s|$ is the length of string s
- ϵ is the empty string, the (unique) string of length 0
- Σ^n is the set of strings of length n
- $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$ is the set of all strings over Σ

Some formalism on strings (1): Examples

Examples

- DNA: $\Sigma = \{\texttt{A},\texttt{C},\texttt{G},\texttt{T}\}$, alphabet size $|\Sigma| = 4$,
  a string of length 5 is $s = \texttt{ACCTG}$, $s_1 = \texttt{A}, s_2 = s_3 = \texttt{C}, s_4 = \texttt{T}, s_5 = \texttt{G}$.
- RNA: $\Sigma = \{\texttt{A},\texttt{C},\texttt{G},\texttt{U}\}$, again alphabet size is 4
- protein: $\Sigma = \{$A,C,D,E,F,...,W,Y$\}$, alphabet size is 20,
  ANRFYWNL is a string over Σ of length 8
- English alphabet: $\Sigma = \{$a,b,c,...,x,y,z$\}$ of size 26

Some formalism on strings (2)

Let $s = s_1 \dots s_n$ be a string over Σ. **ex.** $s = \texttt{ACCTG}$

- t is a substring of s if $t = \epsilon$ or $t = s_i \dots s_j$ for some $1 \leq i \leq j \leq n$
  (i.e., a "contiguous piece" of s) $\texttt{CCT}, \texttt{AC}, \dots$
- t is a prefix of s if $t = \epsilon$ or $t = s_1 \dots s_j$ for some $1 \leq j \leq n$
  (i.e., a "beginning" of s) $\texttt{AC}, \texttt{ACCTG}, \dots$
- t is a suffix of s if $t = \epsilon$ or $t = s_i \dots s_n$ for some $1 \leq i \leq n$
  (i.e., an "end" of s) $\texttt{CCTG}, \texttt{G}, \dots$
- t is a subsequence of s if t can be obtained from s by deleting some
  (possibly 0, possibly all) characters from s $\texttt{ACT}, \texttt{CCT}, \dots$

N.B.
string = sequence, but substring $\neq$ subsequence!

Substrings etc.

N.B.

1. Every substring is a subsequence, but not every subsequence is a substring!
   Ex.: Let $s = \texttt{ACCTG}$, then $\texttt{ACT}$ is a subsequence but not a substring.
2. Every prefix is a substring, every suffix is a substring.
3. t is substring of $s \Leftrightarrow t$ is prefix of a suffix of $s \Leftrightarrow t$ is suffix of a prefix of s

Counting strings

Question
How big is Σ^n, i.e., how many strings of length n are there?

Answer
$|\Sigma^n| = |\Sigma|^n$. E.g. there is $|\Sigma|^0 = 1$ string of length 0, there are 4 strings of length 1 over the DNA alphabet, 16 of length 2, 64 of length 3, etc. (We already saw this argument in connection with the degeneracy of the genetic code.)

Counting substrings, subsequences etc.

Question

Given $s = s_1 \dots s_n$. How many

- prefixes,
- suffixes,
- substrings,
- subsequences

does s have (exactly, at most, at least)?

Formalizing alignments

Informal definition

Given $s, t \in \Sigma^*$ (i.e., s, t are two strings over the same alphabet Σ, not necessarily of the same length), an alignment of s and t is a way of writing one above the other, possibly inserting gaps (denoted "$-$"), in such a way that (a) both have the same length, and (b) no two gaps are above each other.

Ex: five different alignments of $s =$ `ACCT` and $t =$ `CAT`

```
-ACCT    ACCT    ACCT    -ACCT    ---ACCT
CA--T    -CAT    CAT-    CA--T    CAT----
```


Formalizing alignments

Formal definition

An alignment $\mathcal{A}$ of $s, t \in \Sigma^*$ is a matrix with two rows and entries from $\Sigma \cup \{-\}$, where

1. deleting all gaps from the first row yields s
2. deleting all gaps from the second row yields t
3. no column consists of two gaps

Ex:

```
-ACCT    ACCT    ACCT    -ACCT    ---ACCT
CA--T    -CAT    CAT-    CA--T    CAT----
```


Formalizing alignments

Alignment $\mathcal{A}$ has length $|\mathcal{A}|$, and the columns of $\mathcal{A}$ are called $\mathcal{A}^{(i)}$, for $i = 1, \dots, |\mathcal{A}|$.

Ex:

```
-ACCT    ACCT    ACCT    -ACCT    ---ACCT
CA--T    -CAT    CAT-    CA--T    CAT----
```

E.g. for the first alignment above, $\mathcal{A}^{(1)} = \binom{-}{\mathrm{C}}$ and $\mathcal{A}^{(2)} = \binom{\mathrm{A}}{\mathrm{A}}$.

Length of alignments

Given $s, t \in \Sigma^*$ and an alignment $\mathcal{A}$ of s and t, how long is $\mathcal{A}$ at most? At least?

Scoring alignments

Informal definition

The score of an alignment is the sum of the scores of its columns. A scoring function scores each column according to whether it is a match (two characters which are the same), a mismatch (two different characters), or a gap (gap+character or character+gap).

Example

	match	mismatch	gap
f_1	2	-1	-1
f_2	1	-1	-2

Usually match ≥ 0 and mismatch, gap ≤ 0.

Scoring alignments

	match	mismatch	gap
f_1	2	-1	-1

Formal definition

A scoring function f is a pair (p, g), where $p : \Sigma \times \Sigma \to \mathbb{R}$ and $g \in \mathbb{R}$, and for a column $\mathcal{A}^{(i)} = \binom{x}{y}$, we have

$$f\binom{x}{y} = \begin{cases} p(x,y) & \text{if } x, y \in \Sigma \\ g & \text{if } x = - \text{ or } y = -. \end{cases}$$

E.g. for f_1 :

- $g = -1$, and
- $p(a,b) = \begin{cases} 2 & \text{if } a = b \\ -1 & \text{if } a \neq b. \end{cases}$

This will allow us to define more general scoring functions later.

Scoring alignments

So now we have: Given a scoring function $f = (p, g)$ and an alignment $\mathcal{A}$, the score of $\mathcal{A}$ is

$$score(\mathcal{A}) = \sum_{i=1}^{|\mathcal{A}|} f(\mathcal{A}^{(i)}),$$

the sum of the scores of the alignment columns.

Optimal alignments

Def.

Given $s, t \in \Sigma^*$ and scoring function f, the similarity of s and t, is defined as

$$sim(s,t) = \max\{score(\mathcal{A}) \ : \ \mathcal{A} \text{ is an alignment of } s \text{ and } t.\}$$

Optimal alignments

Def.

Given $s, t \in \Sigma^*$ and scoring function f, the similarity of s and t, is defined as

$$sim(s,t) = \max\{score(\mathcal{A}) \ : \ \mathcal{A} \text{ is an alignment of } s \text{ and } t.\}$$

Def.

An optimal alignment of s and t is an alignment $\mathcal{A}$ with maximum score, i.e. an alignment $\mathcal{A}$ s.t.

$$score(\mathcal{A}) = sim(s,t).$$

Equivalently:
$score(\mathcal{A}) = \max\{score(\mathcal{A}') \ : \ \mathcal{A}' \text{ is an alignment of } s \text{ and } t.\}$

Optimal alignments

N.B.

- Whether an alignment is optimal, depends on the scoring function!
- If $\mathcal{A}$ is an optimal alignment of s, t, then, given any alignment $\mathcal{A}'$ of s, t,
$$score(\mathcal{A}) \geq score(\mathcal{A}')$$
(obviously using the same scoring function).
- There may be more than one optimal alignment of two strings s and t.

Our computational problem: Global alignment

Now we can formally state our computational problem:

Problem variant 1

Input: Two strings s, t over alphabet Σ, scoring function f.
Output: An optimal alignment of s and t.

Problem variant 2

Input: Two strings s, t over alphabet Σ, scoring function f.
Output: $sim(s,t)$.

Note that in Variant 2, we want to output a number, we are not interested in an optimal alignment itself.