String-searching Algorithms

Introduction

Sergio Peignier sergio.peignier@insa-lyon.fr

Associate Professor
INSA Lyon
Biosciences department

String-searching algorithms

What is a string?

- Sequence of characters
- Constant/variable
- Data type/structure

What is a character?

- **Unit of information**
- **Object** belonging to a given **alphabet**
- The **alphabet** is the **set of possible characters**

What are the differences between lists, strings and sequences?

- **Sequence**: Enumerated (ordered) collection of objects (repetitions are allowed)
- **List**: Sequence in Computer Sciences (e.g., list of integers...)
- **String**: List of characters

Definitions

- Let $A = \{a_1, a_2, ...\}$ be an alphabet
- Let $S = (s_1, s_2, ... \mid \forall s_i \in A)$ be a string with length $|T|$
- Let $S[i] = s_i$ being the letter at position i in string S
- Let $S[i, j] = (s_i, ...s_j)$ being the **substring** of S between positions i and j
- $S[1, j]$ is a **prefix** of S
- $S[j, n]$ is a **suffix** of S (for $n = |T|$)

Example: DNA string

String-searching **algorithms**

What is an algorithm?

- **Finite** and **unambiguous** sequence of **instructions**, applied to a given **input**, (usually) implementing a given **function** with the aim to solve a given **problem**.

- Usually algorithms are computer-implementable instructions.

How can we characterize an algorithm?

- How well does the algorithm solve the task?
  (e.g., accuracy)

- How much computer time does it take to run?
  Time complexity: count the number of elementary operations needed (assumption: elementary operations take a fixed amount of time)

- How much memory does it take to run?
  Space complexity: count the number of elementary values needed to be stored (variables, constants...)

How can we define an complexity?

- All terms are not equally important while evaluating complexity.
  (example: unique integer value vs. large matrix)

- Complexity is evaluated asymptotically w.r.t. main terms

- The contribution of fast growing terms make the other ones irrelevant (they can be omitted).

- Constants can be omitted.

How can we define an complexity?

Big $\mathcal{O}$ formal notation

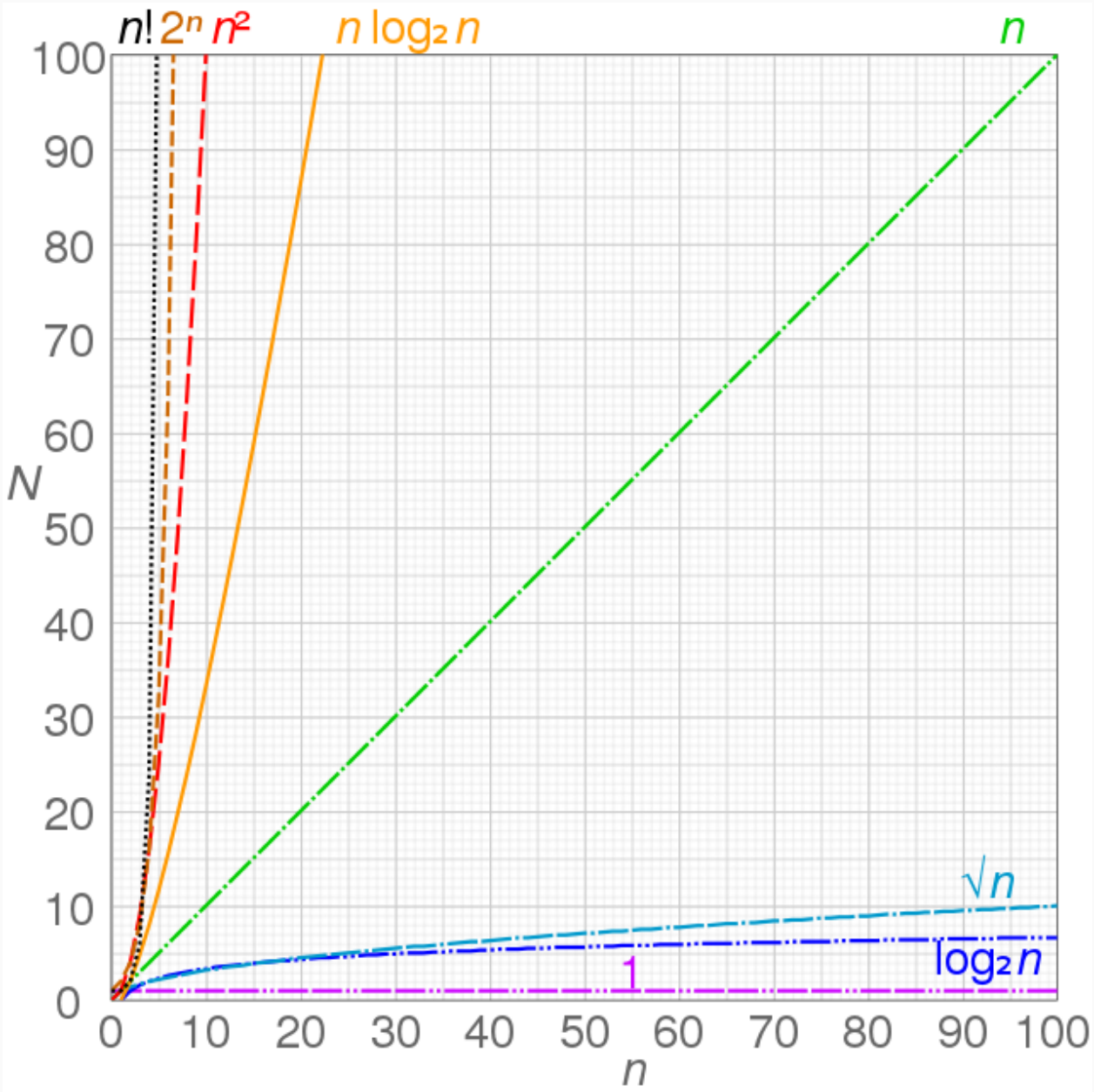
- let $f : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ two functions.
- Let $g(x) > 0 \quad \forall x \geq x^*$
- If $\exists\, M \in \mathbb{R}^+,\ x_0 \in \mathbb{R} \quad | \quad \forall x \geq x_0\ |f(x)| \leq Mg(x)$ then:

$$\boxed{f(x) = \mathcal{O}(g(x))} \quad \text{as } x \to \infty$$

Time complexity examples

	Name	Example
$\mathcal{O}(1)$	Constant	Look-up table
$\mathcal{O}(log\ n)$	Logarithmic	Binary search in sorted array
$\mathcal{O}(n)$	Linear	Argmax in unsorted array
$\mathcal{O}(nlog\ n)$	Linearithmic	Sort array (merge-sort)
$\mathcal{O}(n^2)$	Quadratic	Sort array (bubble sort)
$\mathcal{O}(2^n)$	Exponential	Find all subsets
$\mathcal{O}(n!)$	Factorial	Find all permutations

Complexity examples



Exercise

Write a program that has a $\mathcal{O}(mn^2)$ complexity

String-**searching** algorithms

String-searching problem

- Let $A = \{a_1, a_2, ...\}$ be an alphabet
- Let $\mathcal{S}$ be the set of strings, s.t., $\forall S \in \mathcal{S} \mid \forall s_i \in S \; s_i \in A$
- Let $f : \mathcal{S}^2 \rightarrow \{0, 1\}$ be a **function** that receives two strings $S, T \in \mathcal{S}^2$, and outputs 1 if S is a substring of T and 0 otherwise.
- To do so, S, T should be **compared**
- f could also output the location of instances of S in T.

What does comparing two sequence mean?

- Quantify the **similarity** between two strings
- Define **string metrics** or **string distance functions**
- Different **string metrics** exist:
  - Hamming distance
  - Levenshtein distance (edit distance)
  - TFIDF distance metric
  - ...

Example: Hamming distance metric

- Let $A = \{a_1, a_2, ...\}$ be an alphabet
- Let $\mathcal{S}$ be the set of strings of size n, s.t., $\forall S \in \mathcal{S}$ *and* $\forall s_i \in S$; $s_i \in A$ and $|S| = n$
- Let $d : \mathcal{S}^2 \rightarrow \mathbb{R}^+$, s.t. $d(T, S) = |\{i \ , \quad \forall i \in \{1, ..., |S|\} \mid T[i] \neq S[i]\}|$ be the Hamming distance between T and S, i.e., number of positions where both strings differ.

Example: Hamming distance metric

Does the string similarity depend on the string size? On the vocabulary size?

Let T and S be two random strings of size n s.t. the probabilities of drawing the characters are $i.i.d.$ and equiprobable.

Exercise:

Let us compute the probability $P(d(T, S) = k)$.

Does the string similarity depend on the string size? On the vocabulary size?

$$\forall c \in A \quad P(T[i] = c, S[i] = c) = P(T[i] = c) \times P(S[i] = c)$$
$$= \frac{1}{|A|^2}$$

$$\text{Then: } P(T[i] = S[i]) = \sum_c P(T[i] = c, S[i] = c)$$
$$= \sum_c \frac{1}{|A|^2} = \frac{1}{|A|}$$

$$\text{Therefore: } P(T[i] \neq S[i]) = 1 - \frac{1}{|A|} = \frac{|A|-1}{|A|}$$

Does the string similarity depend on the string size? On the vocabulary size?

$d(S, T) = k$ is the number of locations i s.t. $T[i] \neq S[i]$

$d(S, T) \sim \mathcal{B}(\frac{|A|-1}{|A|}, n)$

Does the string similarity depend on the string size? On the vocabulary size?

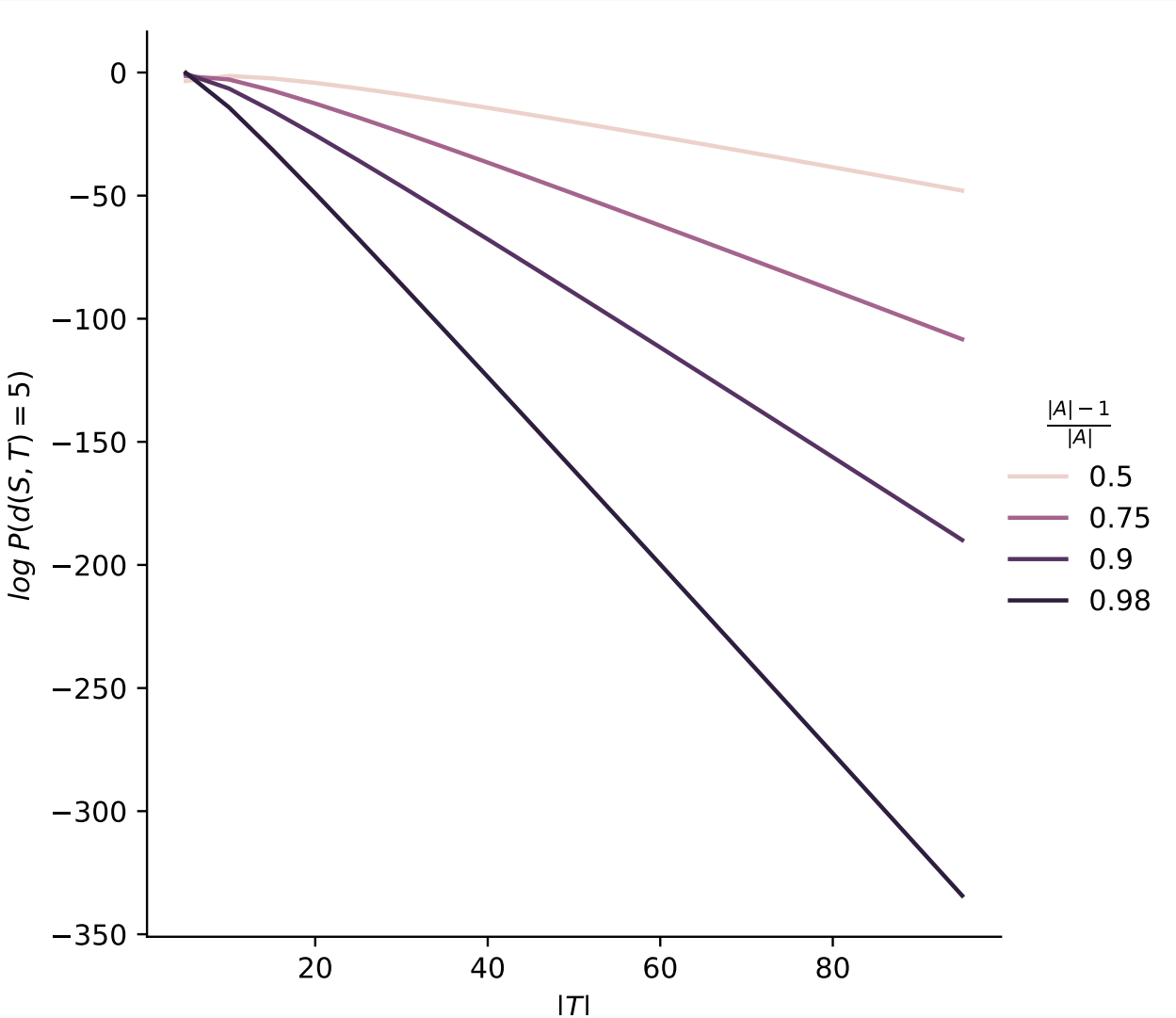
Exercise:

Let us write a python script to compute and plot $P(d(S,T)) = 5$ for $n \in \{5, 10, 15, \ldots, 100\}$ and $|A| \in \{2, 4, 10, 50\}$

Does the string similarity depend on the string size? On the vocabulary size?

```
from scipy.stats import binom
import pandas as pd
k = 5
results = []
for alphabet_size in [2,4,10,50]:
    for sequence_length in range(5,100,5):
        p = 1 - (1 / alphabet_size)
        n = sequence_length
        distribution = binom(n, p)
        prob = distribution.logpmf(k)
        results.append([p,n,prob])
results = pd.DataFrame(results)
```

Does the string similarity depend on the string size? On the vocabulary size?



Why do we need to compare/search strings in Bio-Informatics?

Why do we need to compare/search strings in Bio-Informatics?

Some applications:

- Genome assembly
- Identify mutations (e.g., SNP)
- Determine DNA regions that correlate with phenotypic traits (e.g., QTL)
- Study species evolution (divergence, speciation)
- Establish phylogenetic relationships between species (most recent common ancestor)
- Identify mRNA alternative splicing or editing.
- Quantify gene expression
- Identify functional patterns in protein sequences
- ...

Application examples: Genome Assembly

Genome:

This string is my genome

Reads:

This st

is str

string i

ng is my

s my geno

genome

Application examples: Determine mutations

Genome 1 $\rightarrow$ this stri**n**g is my genome

Genome 2 $\rightarrow$ this stri**m**g is my genome

Application examples: Phylogeny

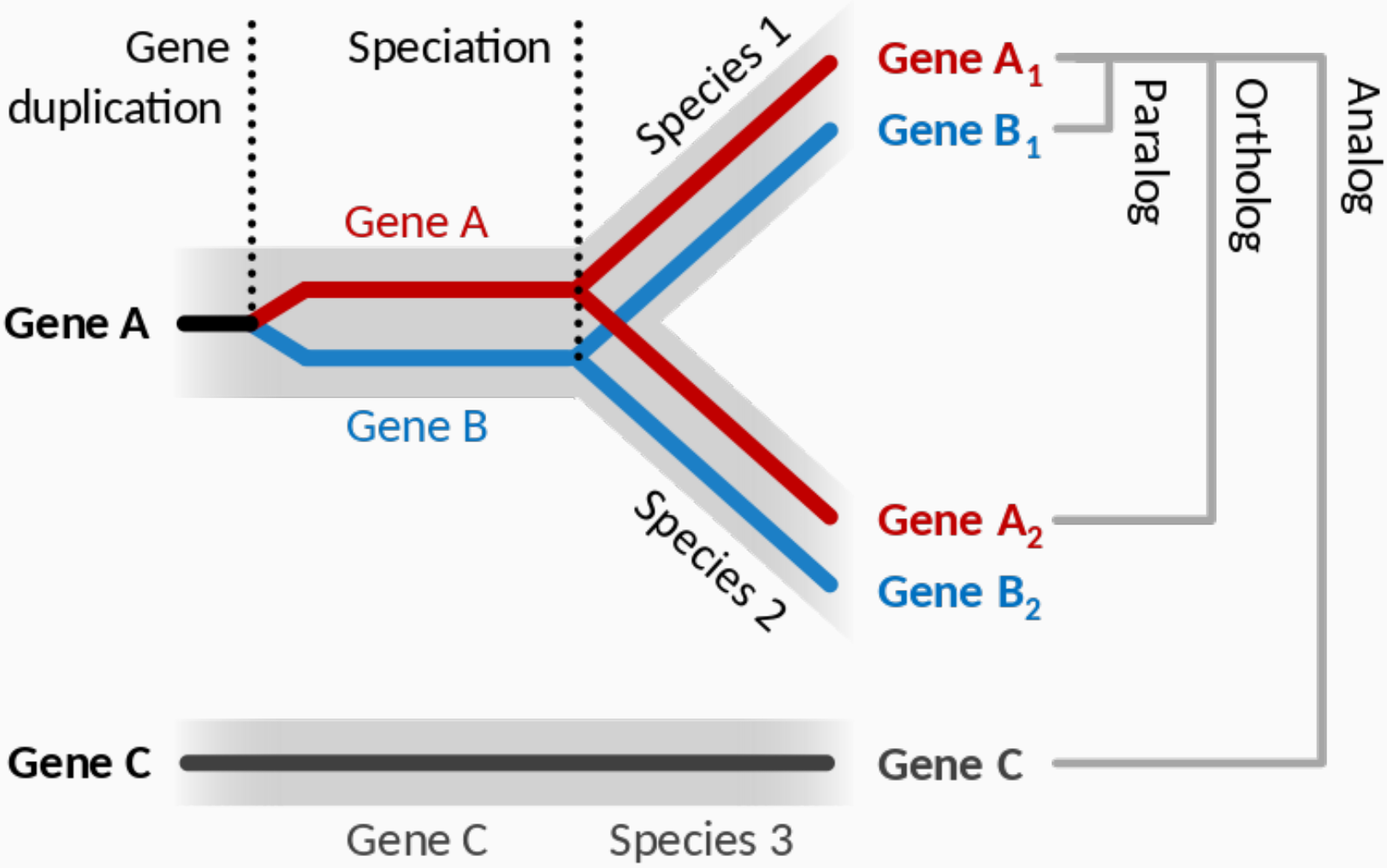
- Given the distance $d(S, T)$ between strings S and T, when can we say that S is similar to T?
  (Ship of Theseus)

- Similarity $\implies$ shared ancestry?

Application examples: Phylogeny

- *S* and *T* are **analogs** if they are **similar** but have **separate evolutionary origins**
- *S* and *T* are **homologous** if their similarity is due to shared ancestry
- **Orthology**: **homology** due to **speciation events**
- **Paralogy**: **homology** due to **duplication events**

Application examples: Phylogeny



Are there other applications for string-searching?

Are there other applications for string-searching?

- Plagiarism checker
- Search engine
- Natural Language Processing