3. Gene prediction

- All genes end on a stop codon
- A simple algorithm for gene prediction
- Searching for start and stop codons
- Predicting all the genes in a sequence
- Making the predictions more reliable
- **Boyer-Moore algorithm**
- Index and suffix trees
- Probabilistic methods
- Benchmarking the prediction methods
- Gene prediction in eukaryotic genomes

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

n = 6

```
AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

m = 32

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
 AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
  AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
   AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
    AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
     AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
      AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

String searching algorithms

- Naive algorithm
  - n: length of the string (or pattern)
  - m: length of the searchable text
  - number of comparisons in the worst case: O (n * (m-n))

```
       AGGAGG
ATAACAGAGGAGGATAATAACAGGAGTAAATA
```

Improving the performance of string searching

- Preprocess the pattern
  - Boyer-Moore algorithm

- Preprocess the searchable text
  - Index
  - Prefix tree

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern

```
CGGCTG
ATAACAGGAGTAAATAACGGCTGGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - no A in the pattern
  - 6 positions (the length of the pattern) can be skipped at once

```
CGGCTG
ATAACAGGAGTAAATAACGGCTGGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - no A in the pattern
  - 6 positions (the length of the pattern) can be skipped at once

```
      CGGCTG
ATAACAGGAGTAAATAACGGCTGGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - no A in the pattern
  - 6 positions (the length of the pattern) can be skipped at once

```
      CGGCTG
ATAACAGGAGTAAATAACGGCTGGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - another C occurs in the pattern
  - slide the pattern accordingly

```
                CGGCTC
ATAACAGGAGTAAATAACGGCTCGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - yet another C occurs in the pattern
  - slide the pattern accordingly

```
              CGGCTC
ATAACAGGAGTAAATAACGGCTCGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern
- A striking example
  - the pattern has been found !

```
                CGGCTC
ATAACAGGAGTAAATAACGGCTCGAGTAAATA
```

The Boyer-Moore algorithm

- Start the comparisons from the end of the pattern
- Use information on the occurrences of characters within the pattern

- Tables of index have to be built from the pattern
- The longer the pattern, the greater is the gain