Mundtlig eksamen for Introduktion til diskret matematik og algoritmer (IDMA) Eksamen 2

Jakob Nordström

Københavns universitet

June 2025

Sorting

Explain how **merge sort** will sort the list

$$[4, 3, 5, 2, 6, 1]$$

Relations

Let R be the relation on $A = \{e_1, e_2, e_3, e_4, e_5\}$ represented by

$$M_R = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(where element e_i corresponds to row and column i).

Can you combine **transitive closure**, **symmetric closure**, **reflexive closure**, and/or **inverse** to obtain the relations below? How?

$$M_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad M_2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad M_3 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Propositional Logic

Decide for each of the propositional logic formulas below whether it is a tautology, contradiction, or neither

1. $(p \to q) \to (q \to p)$
2. $(p \land q) \lor (p \to \neg q)$

Combinatorics

A deck of card has

- 4 **suits:** $\heartsuit$, $\spadesuit$, $\clubsuit$, $\diamondsuit$
- 13 **ranks:** 2–10, jack, queen, king, and ace

A **poker hand** contains 5 cards. Match the three poker hands below with the correct probabilities of drawing them!

(1) Four of a kind: 4 of the cards have the same rank

(2) Full house: 3 cards of one rank, 2 cards of another rank

(3) Flush (including **straight flush**)**:** All cards are of the same suit

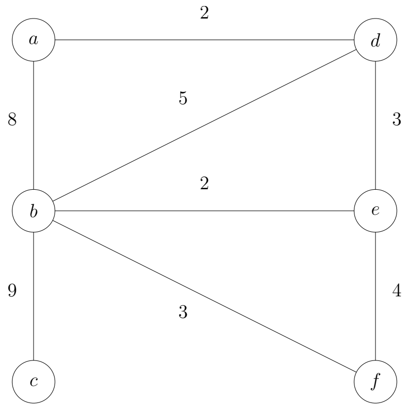
(a) $\dfrac{4 \cdot \binom{13}{5}}{\binom{52}{5}}$ (b) $\dfrac{13 \cdot \binom{4}{3} \cdot 12 \cdot \binom{4}{2}}{\binom{52}{5}}$ (c) $\dfrac{13 \cdot 48}{\binom{52}{5}}$ (d) $\dfrac{\binom{13}{2} \cdot \binom{4}{2} \cdot \binom{4}{2} \cdot 44}{\binom{52}{5}}$

Graph Algorithms

Run the algorithms below on the graph to the right and explain what spanning trees they will produce:

1. Depth-first search
2. Dijkstra's algorithm

(Algorithms that need a starting vertex begin with a; all neighbour lists are sorted in alphabetical order.)



Algorithm Analysis

Below, A and B are arrays indexed from 1 to n containing numbers

```
for i := 1 upto n {
    B[i] := 0
    for j := 1 upto i {
        B[i] := B[i] + A[j]
    }
}
```

1. Explain in plain language what the algorithm does
2. Provide an asymptotic analysis of the running time
3. Can you improve the code to run faster while retaining the same functionality?