



Proseminar *Datenstrukturen und Algorithmen*

Einführung

Wintersemester 2026/27, 15. Oktober 2026

Thomas Noll, Darion Haase, Daniel Zilken

Software Modeling and Verification Group

RWTH Aachen University

<https://moves.rwth-aachen.de/teaching/ws-2026-27/dsal/>

Übersicht

Einführung

Termine

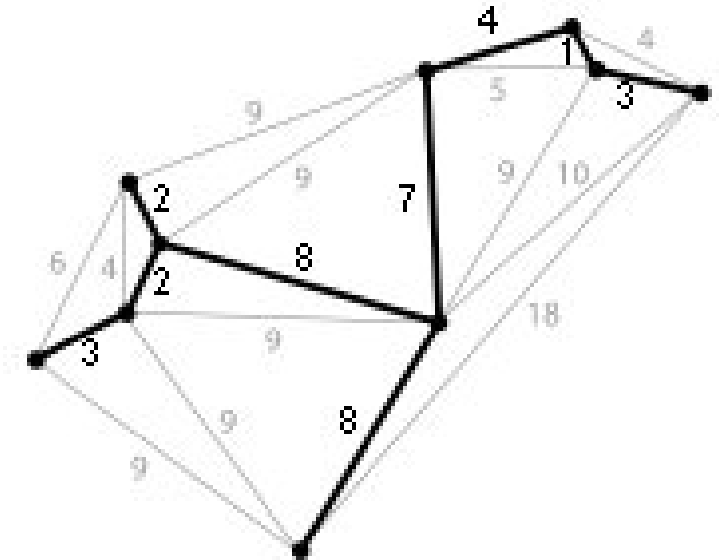
Algorithmen

Datenstrukturen

Thema des Proseminars

Weiterführung und Vertiefung diverser Themen der Vorlesung
Datenstrukturen und Algorithmen

- Algorithmen:
 - neue/neuartige Algorithmen
 - andere Komplexitätsmaße, ...
- Datenstrukturen:
 - Bäume
 - Hashing, ...
- Inhalt:
 - Problemstellung
 - Arbeitsweise des Algorithmus bzw. der Datenstruktur
 - Effizienzeigenschaften
 - Anwendungen



Ziele des Proseminars

- Selbstständiges Einarbeiten in ein neues Thema
- Literaturrecherche
 - Einstiegsreferenz auf Webseite
 - Weitere Literatur online und in Informatikbibliothek
- **Verständliches Präsentieren**
- Kurzdarstellung des Inhalts in einer **wissenschaftlich orientierten Ausarbeitung**

Vortrag

- 20-minütiger Vortrag
- Zielgruppengerechte Präsentation der Inhalte
- übersichtliche Folien:
 - ≤ 15 Textzeilen
 - sinnvoller Einsatz von Farben
- $\text{\LaTeX}$ /beamer-Vorlage wird zur Verfügung gestellt
- Vortrag in Deutsch oder Englisch

Ausarbeitung

- Selbstständiges Verfassen einer Zusammenfassung von gut **5 Seiten**
- **Vollständiges** Literaturverzeichnis
- Korrektes **Zitieren**
- **Plagiarismus:**
Die nicht gekennzeichnete Übernahme fremder Inhalte (Literatur, Web, KI-Tools) führt zum **sofortigen Ausschluss** – dies gilt auch für LLMs!
- Schriftgröße **12pt**, übliche Seitenränder
- **Titelseite** mit Thema, Titel Proseminar, Semester, Name, Datum
- **L^AT_EX-Vorlage** wird zur Verfügung gestellt
- **Sprache** Deutsch oder Englisch
- **Korrekte Sprache** wird vorausgesetzt

Übersicht

Einführung

Termine

Algorithmen

Datenstrukturen

Themenauswahl

Verfahren

- Themenliste wurde/wird ausgehändigt
- Priorisierte Auswahl
- Abgabe hier oder bis (spätestens) **Montag, 19. Oktober per Mail** (noll@cs.rwth-aachen.de)
- Wir bemühen uns (ohne Garantie) um ein „optimales“ Matching
- Zuordnung der Themen und Betreuer:innen bis Mitte kommender Woche online

Rücktritt vom Proseminar

- Bis zu **einer Woche (!)** nach Themenvergabe: ohne Folgen
- Danach: Fehlversuch

Deadlines

Folgende Termine sind **verpflichtend**:

- 19. Oktober: Frist für Themenauswahl
- 28. Oktober: letzte Rücktrittsmöglichkeit
- 16. November: Vorlage des detaillierten Vortragsstruktur
 - nicht: „1. Einleitung/2. Hauptteil/3. Schluss“
 - sondern: detaillierte Strukturübersicht (Einteilung in Abschnitte, wesentliche Definitionen/Aussagen/Beispiele, ...)
- 14. Dezember: vollständige Fassung der Vortragsfolien
- 11.–14. Januar: Blockseminar (halber Tag; wird noch abgestimmt)
- 1. Februar: Einreichung der Ausarbeitung

Übersicht

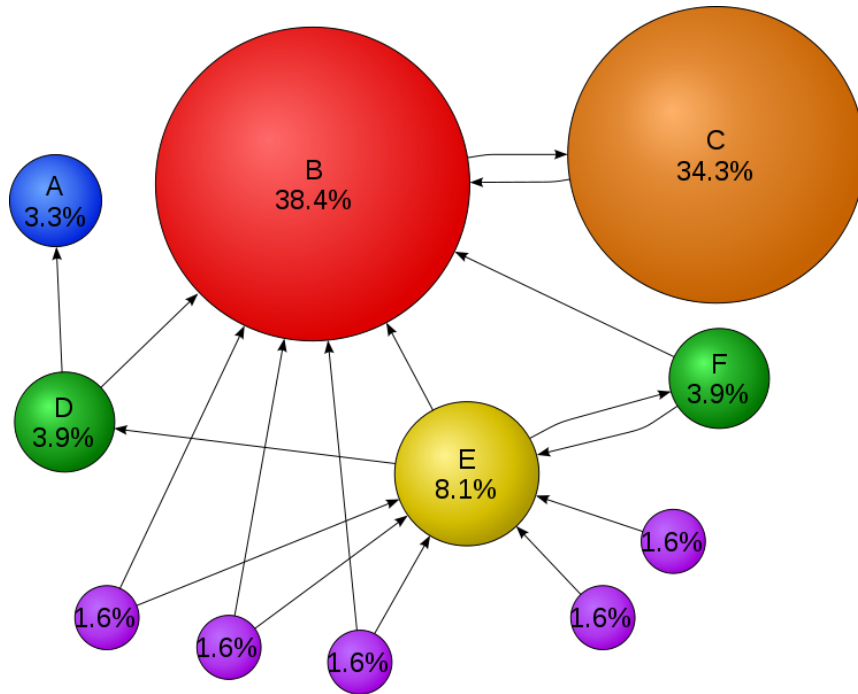
Einführung

Termine

Algorithmen

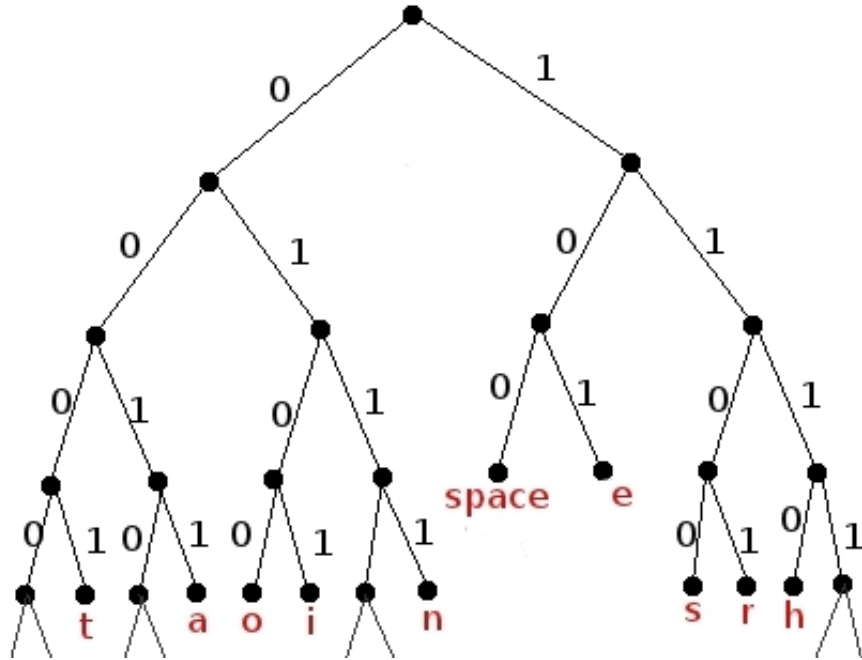
Datenstrukturen

1. Pagerank-Algorithmus



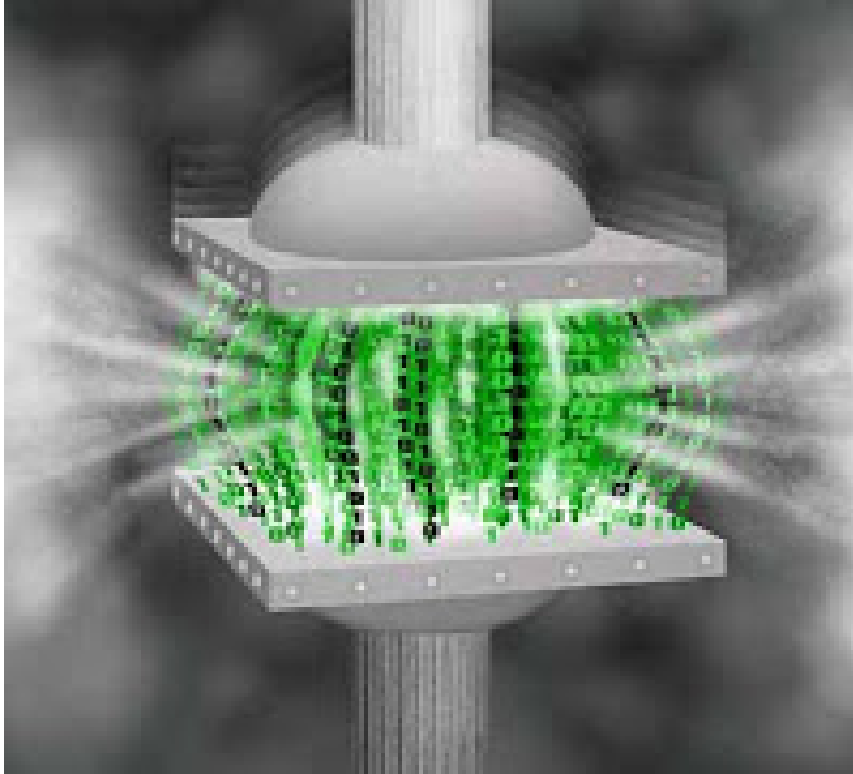
- Algorithmus zur Analyse verlinkter Webseiten
- Allgemeiner: beliebige Graphen
- Ansatz: Bestimmung der Wahrscheinlichkeit, dass zufälliges Benutzerverhalten auf die Seite führt („random surfer“)
- Benutzung in Google-Suchmaschine

2. Huffman-Kodierung



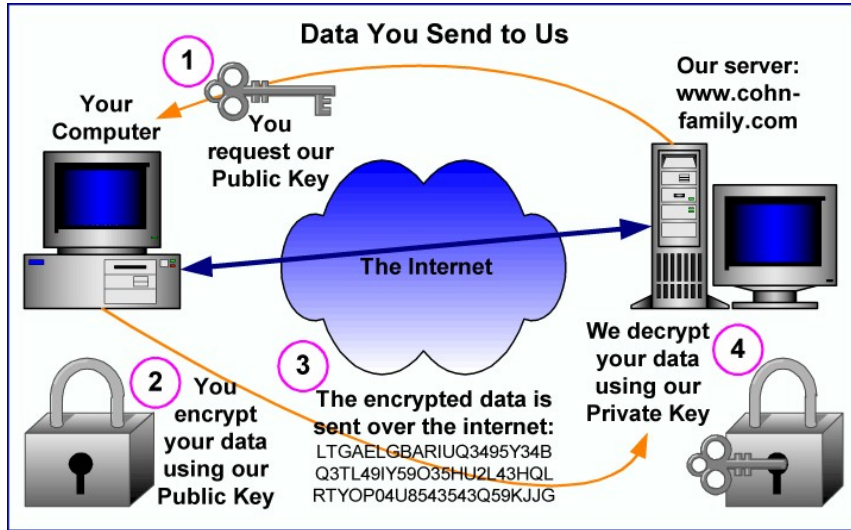
- Ziel: Erstellung einer Binärkodierung mit minimaler mittlerer Wortlänge benötigt
- Basis: Häufigkeitsverteilung der einzelnen Zeichen
- Ansatz:
 - Verwendung eines voll verzweigten Binärbaums zur Darstellung des Codes
 - Kurze Pfadlängen für häufige Zeichen

3. LZW-Datenkompression



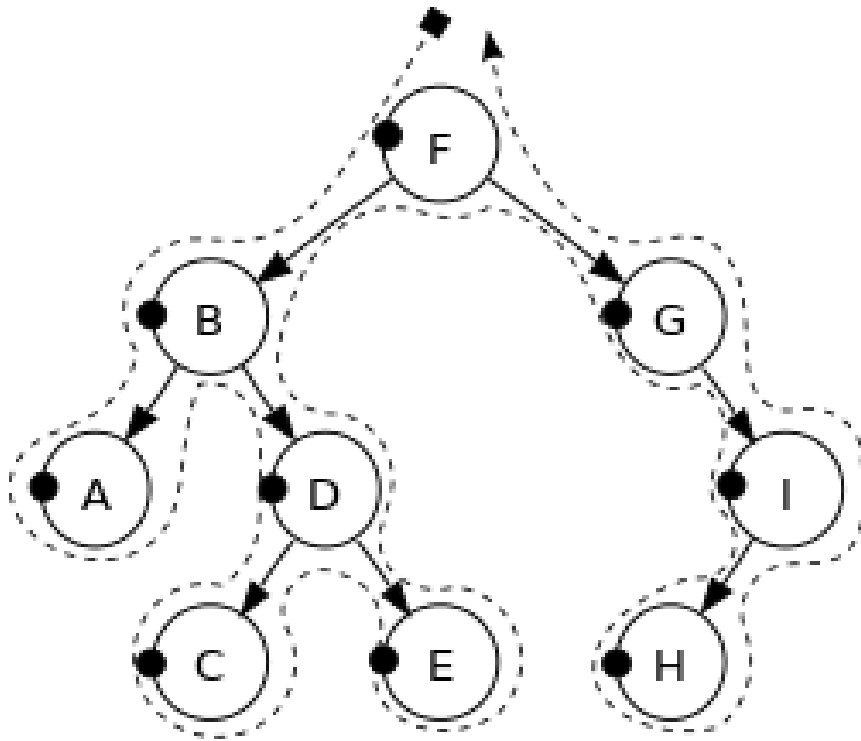
- Lempel-Ziv-Welch-Algorithmus
- Häufig bei Grafikformaten eingesetzt
- Verlustfrei
- Kompression mittels Wörterbüchern mit am häufigsten vorkommenden Zeichenketten
- Wörterbuch nicht zusätzlich gespeichert, sondern aus Datenstrom rekonstruiert

4. RSA-Kryptosystem



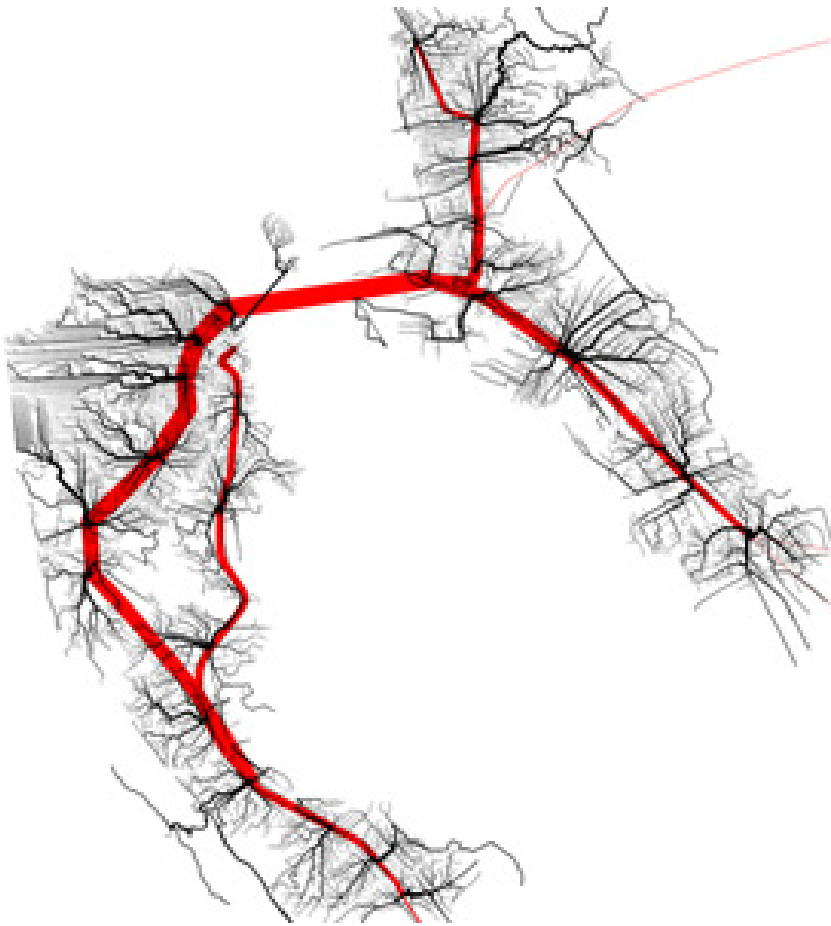
- Verfahren von Rivest, Shamir und Adleman („public-key encryption“)
- Verwendung zur Verschlüsselung und digitalen Signatur
- Asymmetrisch: Schlüsselpaar privat/öffentlich
 - privat: zum Entschlüsseln oder Signieren
 - öffentlich: zur Verschlüsselung/ Signaturprüfung
- Privater Schlüssel wird geheim gehalten und kann nur mit extrem hohem Aufwand aus dem öffentlichen Schlüssel berechnet werden

5. Deutsch-Schorr-Waite Baumtraversierung



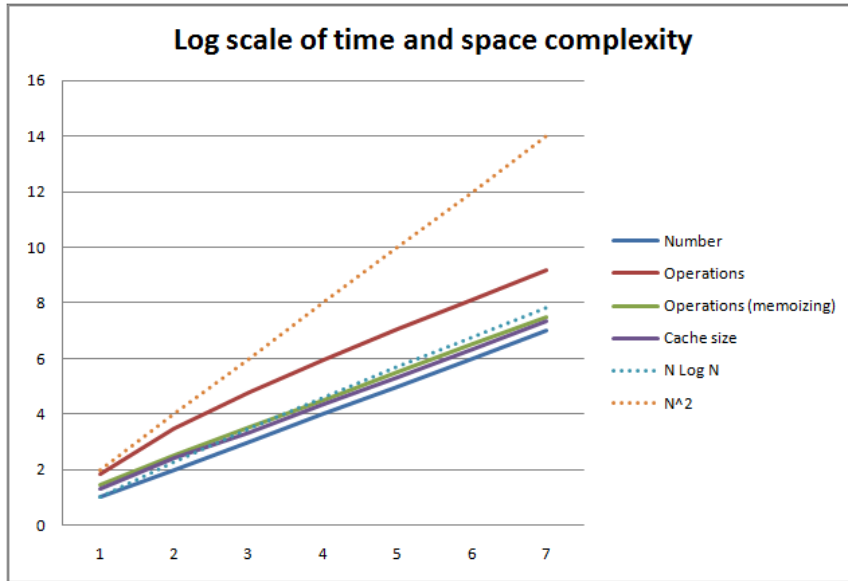
- Klassische Lösung: Rekursion/Stack
- Ziel: Vermeidung des zusätzlichen Speicheraufwands
- Ansatz: systematische Pointerrotation

6. Problem der K kürzesten Pfade



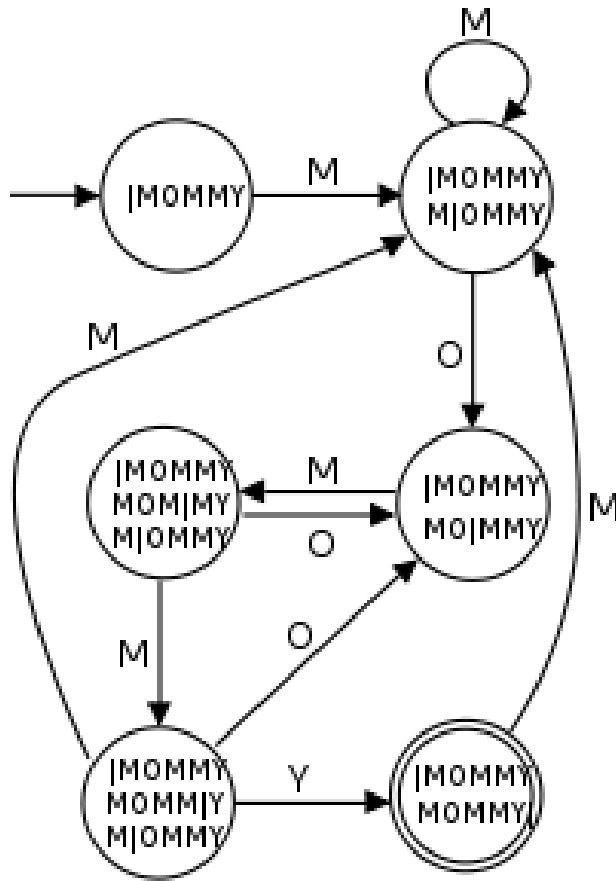
- Problem: Finden der K kürzesten Pfade zwischen Knotenpaar in einem gerichteten Graphen
- Ansatz: Verallgemeinerung der Bellman-Gleichung

7. Amortisierte Laufzeitanalyse



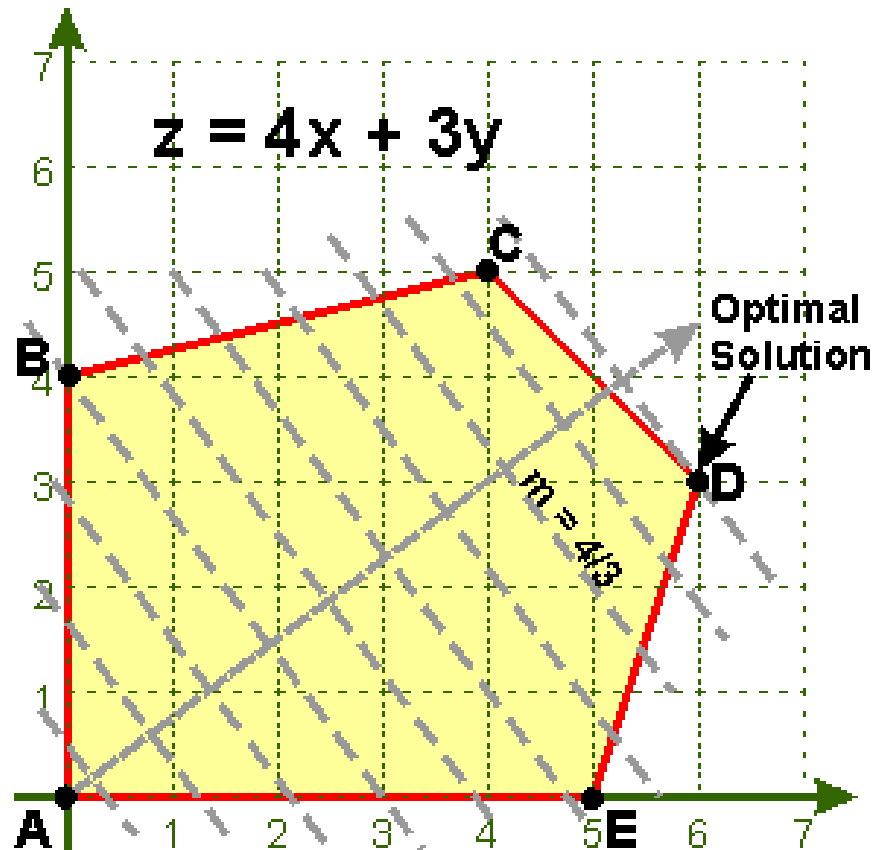
- Allgemeine Laufzeitanalyse: maximale Kosten der einzelnen Schritte
- Amortisierte Laufzeitanalyse: Worst Case aller Operationen im gesamten Durchlauf des Algorithmus
- Verbesserung der oberen Schranke bei seltenem Auftreten teurer Operationen
- Prinzip: der Worst Case ändert den Zustand der Datenstruktur so ab, dass er längere Zeit nicht mehr auftreten kann (z.B. dynamische Arrays)
- Drei unterschiedliche Berechnungsmethoden (Aggregat-, Account-, Potentialfunktion-Methode)

8. String Matching



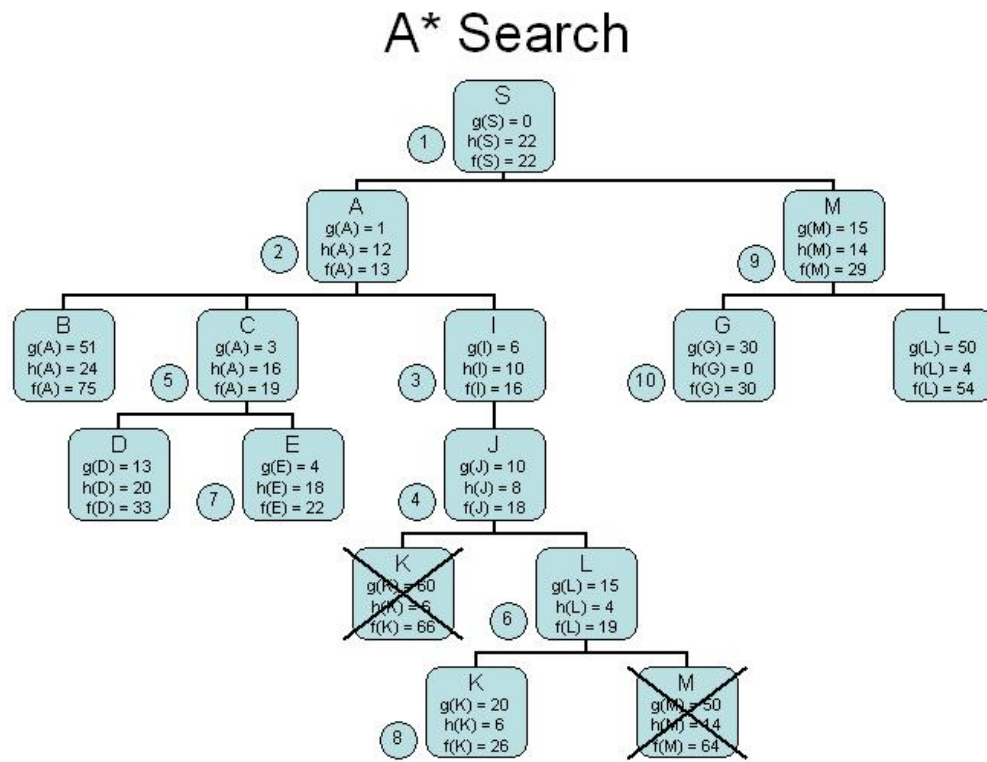
- Ziel: Suche von Vorkommen eines Strings in einem (großen) Text
- Komplexität des naiven Algorithmus: $|\text{Text}| \cdot |\text{String}|$
- Verbesserung durch Vorverarbeitung des Suchstrings

9. Lineare Programmierung (aka Lineare Optimierung)



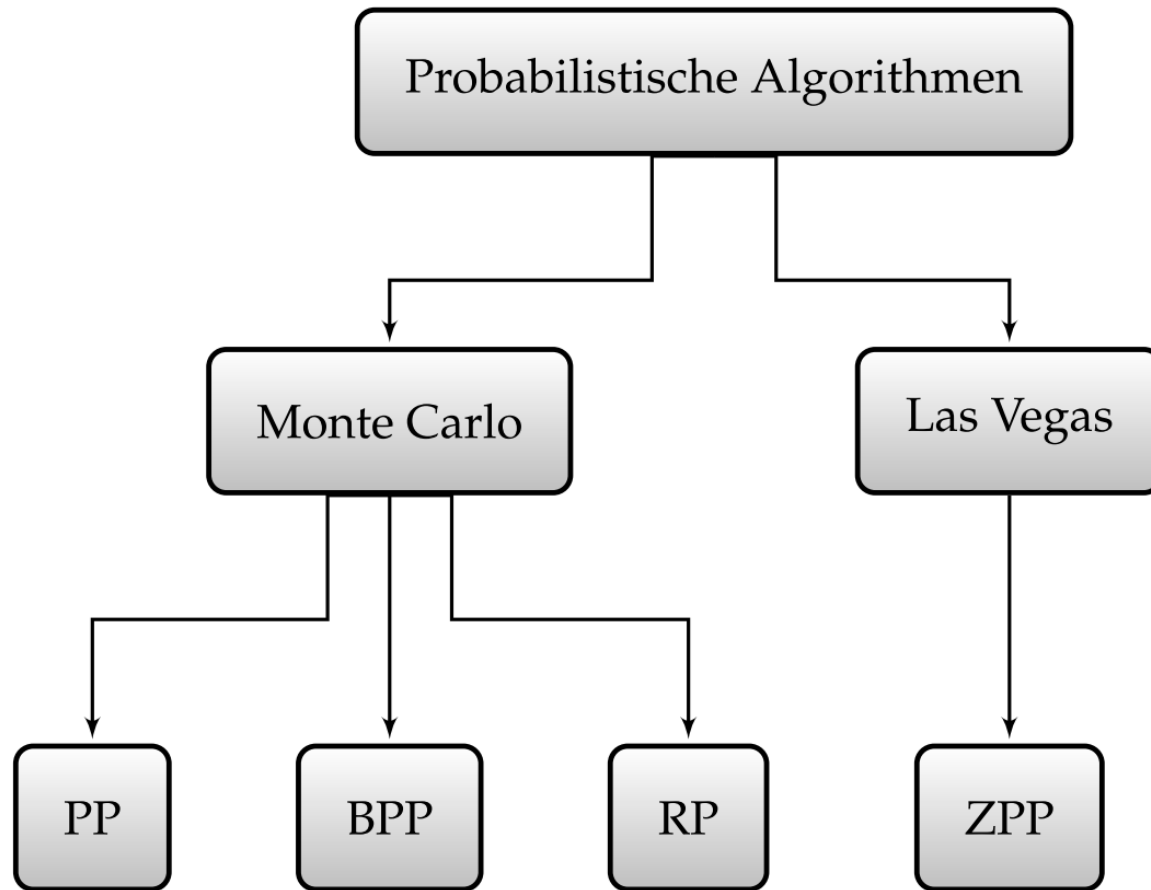
- Ziel: Optimierung linearer Zielfunktionen über einer Menge, die durch lineare (Un-)Gleichungen eingeschränkt ist
- Simplexverfahren
- Ellipsoid-Methode

10. A*-Suchalgorithmus



- Berechnung eines kürzesten Pfades zwischen zwei Knoten in einem Graphen mit positiven Kantengewichten
- Verallgemeinerung und Erweiterung des Dijkstra-Algorithmus
- Kostenfunktion (abschätzende Heuristik) zur zielgerichteten Suche

11. Randomisierte Algorithmen



- Ziel: Verbesserung der Effizienz deterministischer Algorithmen durch Zufallselement
- Monte-Carlo-Algorithmen:
 - können falsche Resultate liefern (mit geringer Wahrscheinlichkeit)
 - deterministische Ausführungszeit
 - Beispiel: Minimaler Cut eines Graphen
- Las-Vegas-Algorithmen:
 - liefern immer korrektes Resultat
 - Ausführungszeit schwankt
 - Beispiel: Quicksort mit randomisierter Auswahl des Pivotelements

Übersicht

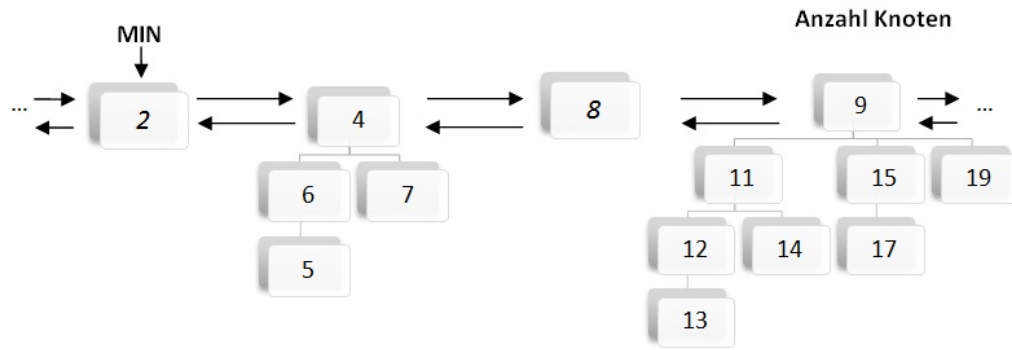
Einführung

Termine

Algorithmen

Datenstrukturen

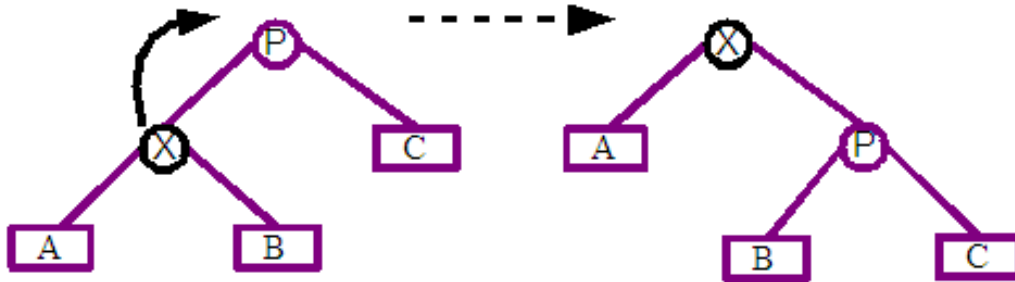
12. Fibonacci-Heaps



- Datenstruktur zur Implementierung einer Vorrangwarteschlange (*priority queue*)
- Liste von geordneten Bäumen
- Heap-Bedingung: Priorität jedes Knotens mindestens so groß wie Priorität seiner Kinder
- (Fast) alle Operationen haben amortisiert konstante Laufzeit

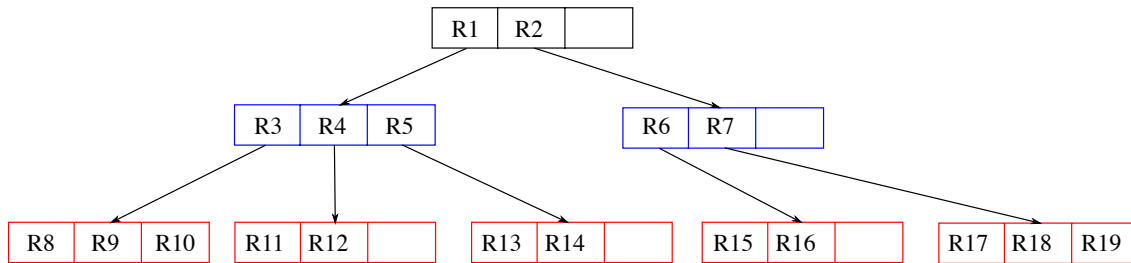
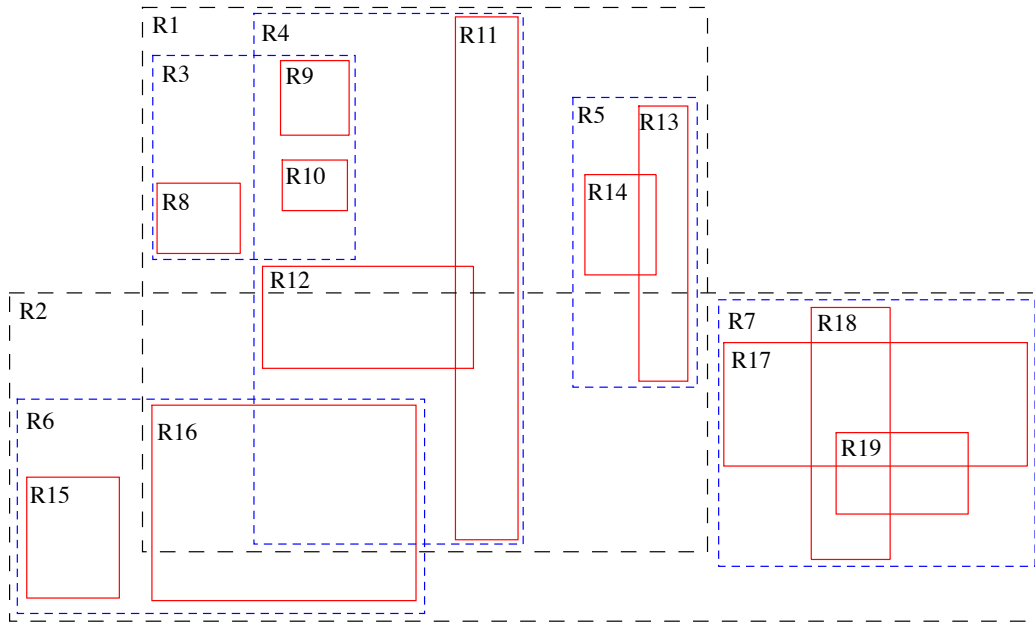
13. Splay-Bäume

Rotate Right



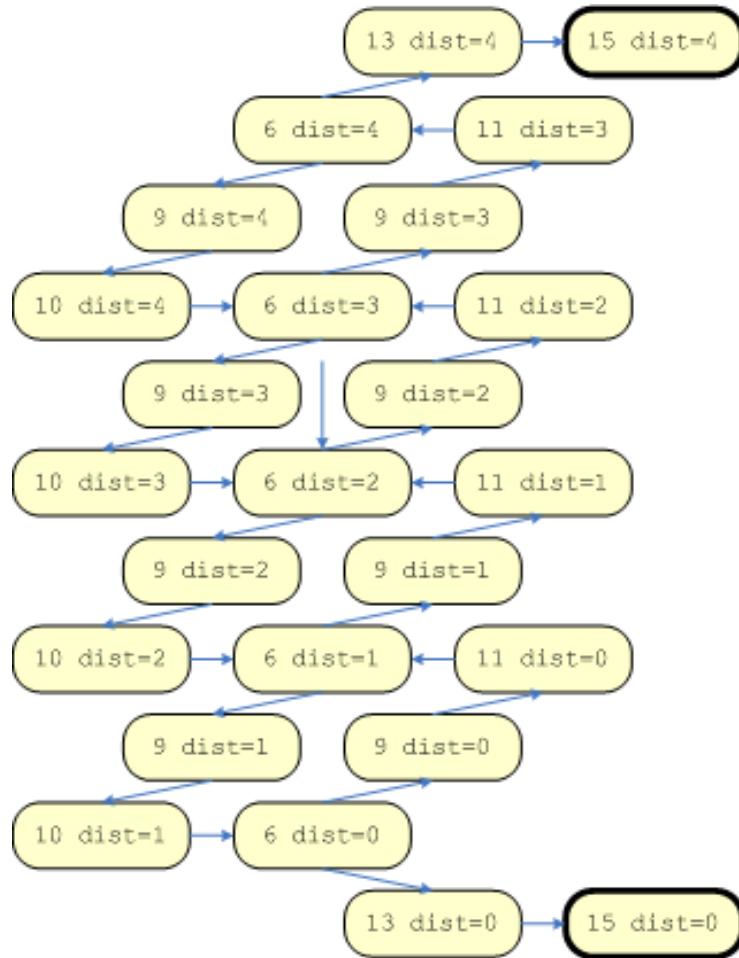
- Baumorganisierte Datenstruktur
- Effizientes Einfügen, Suchen und Löschen von Elementen
- Ansatz: häufig benutzte Einträge nah an Wurzel platzieren
- Grundlegende Operation: *splay* (Rotieren eines Elements bis zur Wurzel)

14. R-Bäume



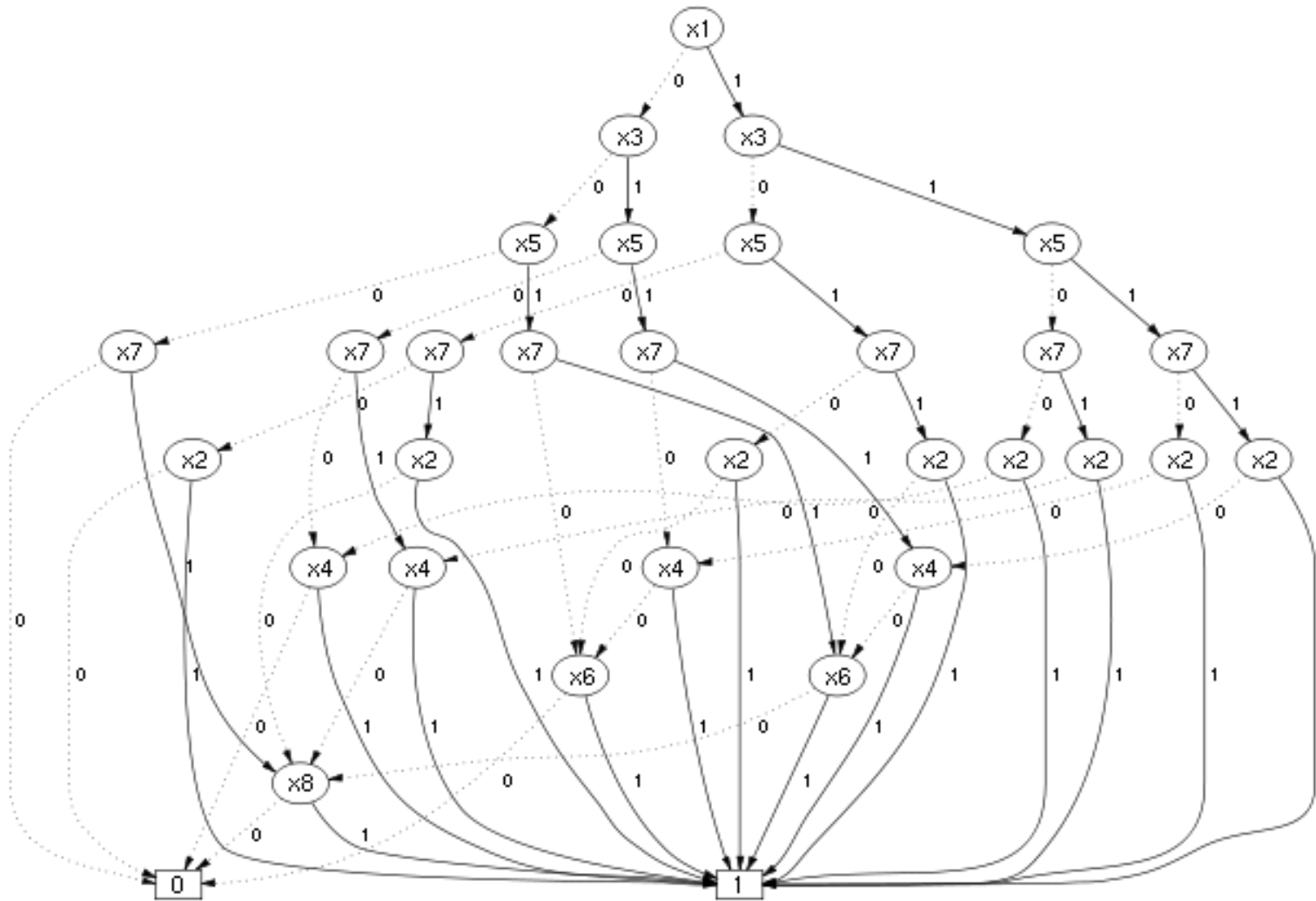
- Räumliche, dynamische, balancierte Indexstruktur
- Für schnelle Suche in mehrdimensionalen ausgedehnten Objekten
- Typisches Einsatzgebiet: Geodatenbanken

15. Bitstate Hashing



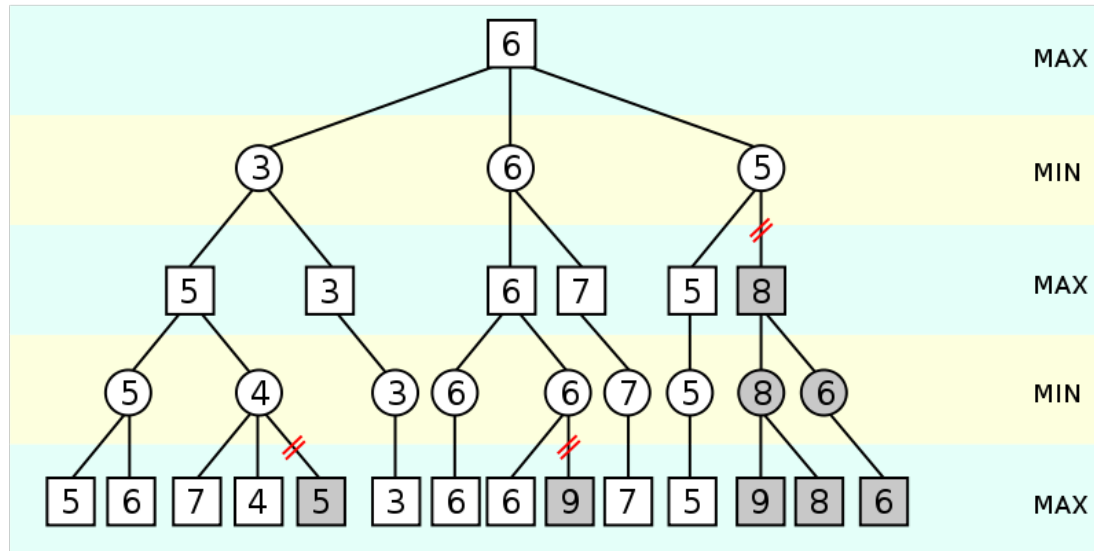
- Effiziente Darstellung von Zustandsräumen
- Problem: Erkennung von Zykeln
- Idee: Benutzung einer Hashfunktion, Speicherung von 0/1 im Hasharray
- Bis zu 98% Speicherersparnis im SPIN-Tool

16. Binäre Entscheidungsdiagramme



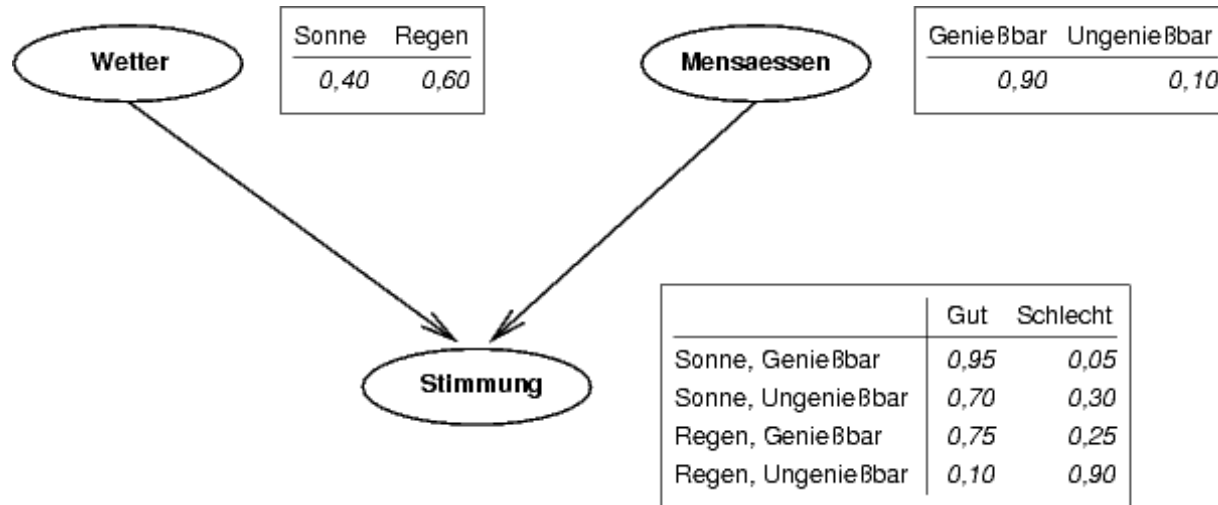
- Datenstruktur zur Darstellung Boolescher Funktionen über mehreren Variablen
- Einsatz: Verifikation von (Hardware-)Systemen
- Effizient durch Sharing gemeinsamer Teilbäume
- Problem: Finden geeigneter Variablenordnungen

17. Spielbäume



- Darstellung von 2-Personen-Spielen mit abwechselnden Zügen (z.B. Schach, Go, Reversi, Dame, Mühle oder Vier gewinnt)
- Ziel: Bestimmung optimaler Strategien
- Minimax-Verfahren
- α - β -Pruning

18. Bayessche Netze



- Gerichteter azyklischer Graph (DAG)
 - Knoten = Zufallsvariablen
 - Kanten = bedingte Abhängigkeiten
- Kompakte Darstellung der gemeinsamen Wahrscheinlichkeitsverteilung aller Variablen
- Algorithmen:
 - Logisches Schließen (Inferenz)
 - Lernen