Proseminar *Datenstrukturen und Algorithmen*

Einführung

Sommersemester 2024, 15. April 2024

Thomas Noll et al.
Software Modeling and Verification Group
RWTH Aachen University

`https://moves.rwth-aachen.de/teaching/ss-24/dsal/`

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

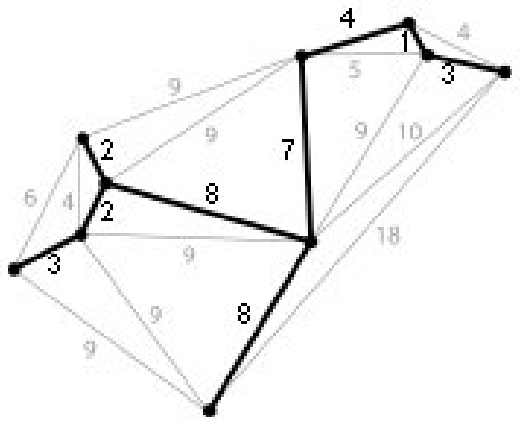
Übersicht

Seminarthema

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



Zielsetzung

Ziele des Proseminars

- Selbstständiges Einarbeiten in ein neues Thema
- Literaturrecherche
  - Einstiegsreferenz auf Webseite
  - Schulung in Informatikbibliothek
- Verständliches Präsentieren
- Kurzdarstellung des Inhalts in einer wissenschaftlich orientierten Ausarbeitung

Bearbeitung in Zweiergruppen

- Zwei separate Vorträge (mit nicht notwendigerweise verschiedenen Foliensätzen)
- Gemeinsame Anfertigung der Ausarbeitung

Anforderungen Vortrag

Vortrag

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

Anforderungen Ausarbeitung

Ausarbeitung

- Selbstständiges Verfassen einer Zusammenfassung von gut 5 Seiten
- Vollständiges Literaturverzeichnis
- Korrektes Zitieren
- Plagiarismus:
  Die nicht gekennzeichnete übernahme fremder Inhalte führt zum sofortigen Ausschluss.
- Schriftgröße 12pt, übliche Seitenränder
- Titelseite mit Thema, Titel Proseminar, Semester, Name, Datum
- LaTeX-Vorlage wird zur Verfügung gestellt
- Sprache Deutsch oder Englisch
- Korrekte Sprache wird vorausgesetzt

Übersicht

Themenauswahl

Verfahren

- Themenliste wurde/wird ausgehändigt
- Priorisierte Auswahl
- ggf. Angabe Wunschpartner(in)
- Abgabe bis (spätestens) Freitag, 19. April per Mail/im Sekretariat
- 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 drei Wochen nach Themenvergabe: ohne Folgen
- Danach: Fehlversuch

Bibliothekseinführung

Einführung in die Literaturrecherche

- Einweisung in themenspezifische Literaturrecherche
- Dauer: ca. zwei Stunden
- Teilnahme für BSc-Studierende verpflichtend
- Bedarf bitte auf Themenblatt vermerken
- Termine zur Auswahl:
  - Mittwoch, 17.04.2024, 10 Uhr
  - Donnerstag, 18.04.2024, 14 Uhr
  - Mittwoch, 24.04.2024 10 Uhr
- Mögliche Termine auf Liste vermerken, Bestätigung per Mail

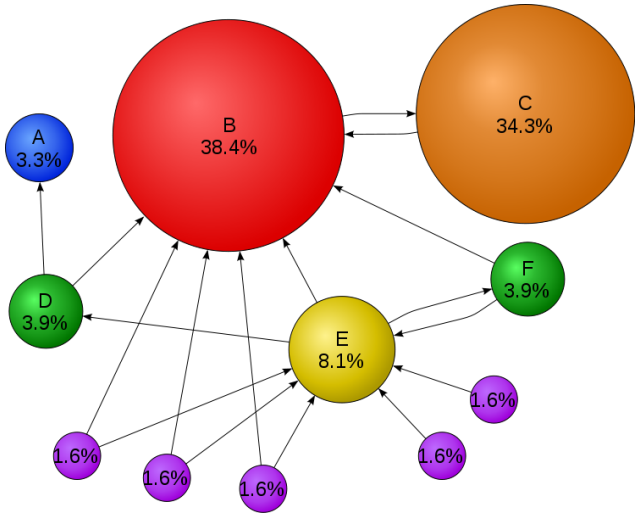
Deadlines

Folgende Termine sind verpflichtend:

- 19. April: Frist für Themenauswahl
- 10. Mai: letzte Rücktrittsmöglichkeit
- 13. Mai: Vorlage des detaillierten Vortragsstruktur
  - nicht: „1. Einleitung/2. Hauptteil/3. Schluss“
  - sondern: detaillierte Strukturübersicht (Einteilung in Abschnitte, wesentliche Definitionen/Aussagen/Beispiele, ...)
- 10. Juni: vollständige Fassung der Vortragsfolien
- 08./09./10. Juli (?): Blockseminar
- 22. Juli: Einreichung der Ausarbeitung

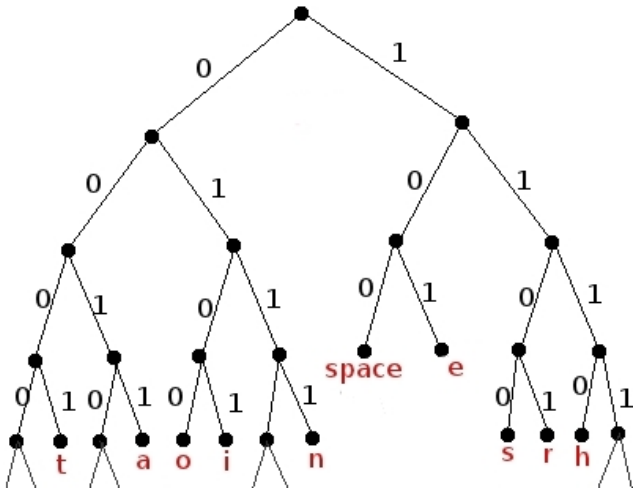
Übersicht

1. Pagerank-Algorithmus



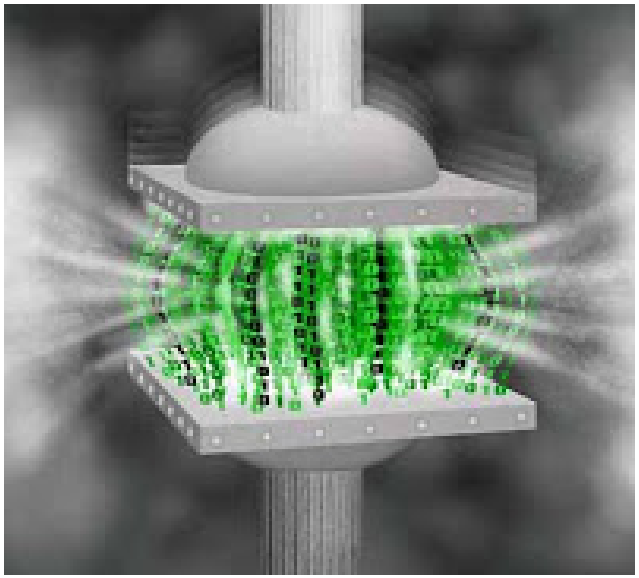
- Algorithmus zur Analyse verlinkter Webseiten
- Allgemeiner: beliebige Graphen
- Ansatz: Bestimmung der Wahrscheinlichkeit, dass zufälliges Benutzerverhalten auf die Seite führt („random surfer“)
- Benutzung in Google-Suchmaschine

2. Huffman-Kodierung



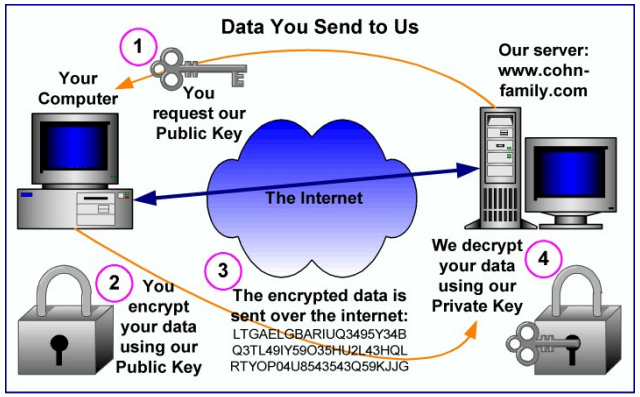
- Ziel: Erstellung einer Binärkodierung mit minimaler mittlerer Wortlänge benötigt
- Basis: Wahrscheinlichkeitsverteilung der einzelnen Zeichen
- Ansatz:
  - Verwendung eines voll verzweigten Binärbaums zur Darstellung des Codes
  - Kurze Pfadlängen für häufige Zeichen

3. LZW-Datenkompression



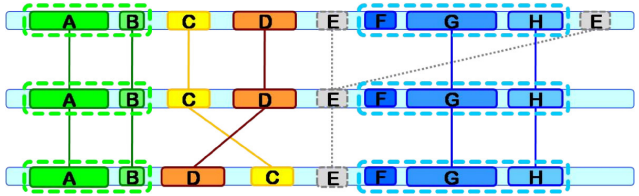
- Lempel-Ziv-Welch-Algorithmus
- Häufig bei Grafikformaten eingesetzt
- Verlustfrei
- Kompression mittels Wörterbüchern mit am häufigsten vorkommenden Zeichenketten
- Wörterbuch nicht zusätzlich gespeichert, sondern aus Datenstrom rekonstruiert

4. RSA-Kryptosystem



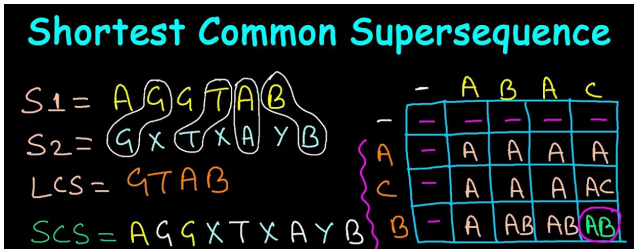
- Verfahren von Rivest, Shamir und Adleman („public-key encryption“)
- Verwendung zur Verschlüsselung und digitalen Signatur
- Asymmetrisch: Schlüsselpaar privat/öffentlich
  - privat: zum Entschlüsseln oder Signieren
  - öffentlich: zur Verschlüsselung/ Signaturprüfung
- Privater Schlüssel wird geheim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

5. Längste gemeinsame Teilsequenz



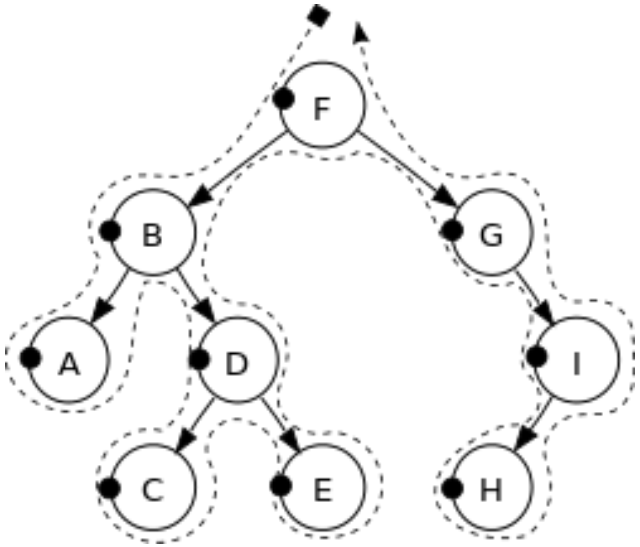
- Definition: längste gemeinsame (nicht notwendigerweise zusammenhängende) Teilsequenz mehrerer Zeichenketten
- Anwendungen: `diff`, Bioinformatik (Genanalyse)
- NP-hart für beliebige Anzahl von Zeichenketten
- Quadratische Komplexität für zwei Zeichenketten (dynamische Programmierung)

6. Kürzeste gemeinsame Obersequenz



- Aufgabe: Gegeben mehrere Zeichenketten, finde die kürzeste Zeichenfolge, welche alle Ketten als Teilsequenzen enthält
- NP-hartes Problem
- Effiziente approximative Algorithmen

7. Deutsch-Schorr-Waite Baumtraversierung

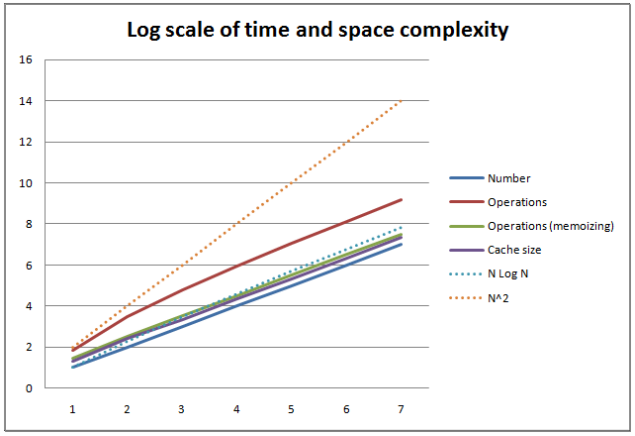


- Klassische Lösung: Rekursion/Stack
- Ziel: Vermeidung des zusätzlichen Speicheraufwands
- Ansatz: systematische Pointerrotation

8. Problem der K kürzesten Pfade

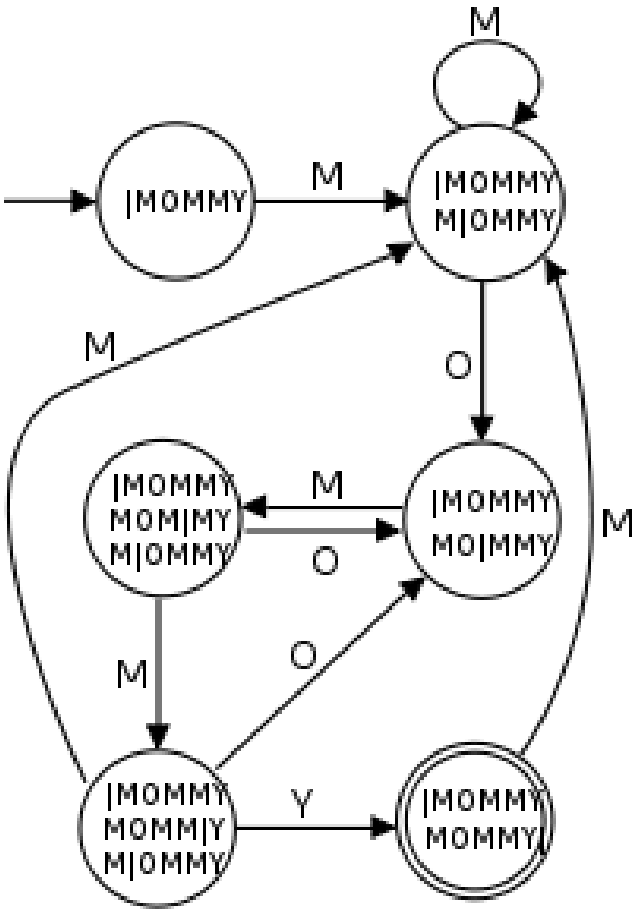
- Problem: Finden der K kürzesten Pfade zwischen Knotenpaar in einem gerichteten Graphen
- Ansatz: Verallgemeinerung der Bellman-Gleichung

9. Amortisierte Laufzeitanalyse



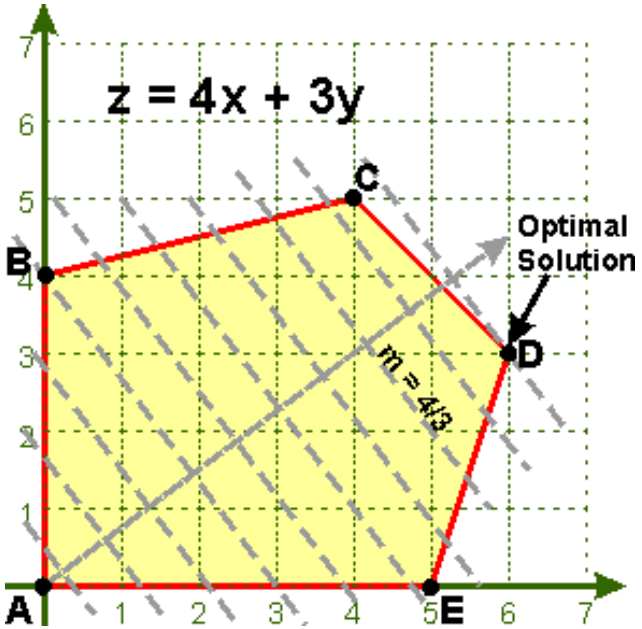
- Allgemeine Laufzeitanalyse: maximale Kosten der einzelnen Schritte
- Amortisierte Laufzeitanalyse: Worst Case aller Operationen im gesamten Durchlauf des Algorithmus
- Verbesserung der oberen Schranke bei seltenem Auftreten teurer Operationen
- Idee: der Worst Case ändert den Zustand der Datenstruktur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

10. String Matching



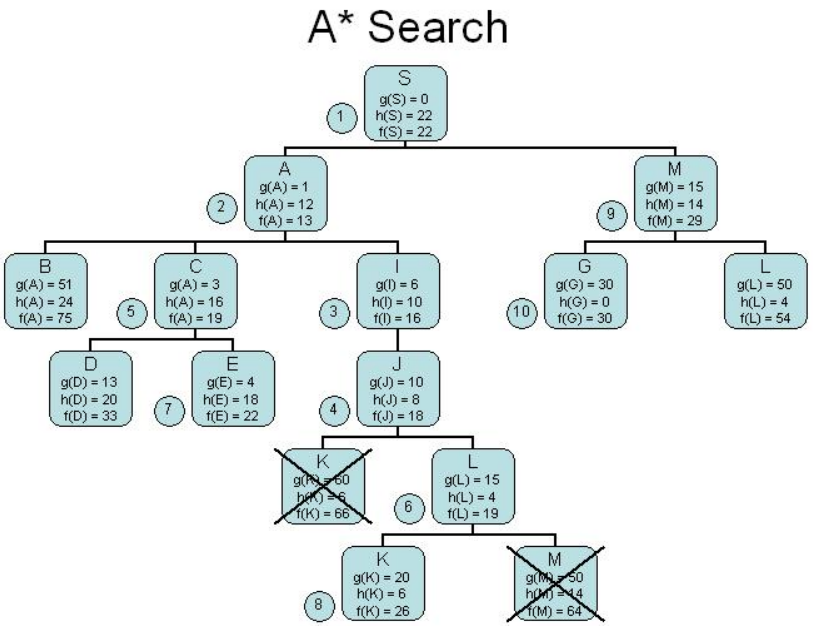
- Ziel: Suche von Vorkommen eines Strings in einem (großen) Text
- Komplexität des naiven Algorithmus: $|\text{Text}| \cdot |\text{String}|$
- Verbesserung durch Vorverarbeitung des Suchstrings

11. Lineare Programmierung (aka Lineare Optimierung)



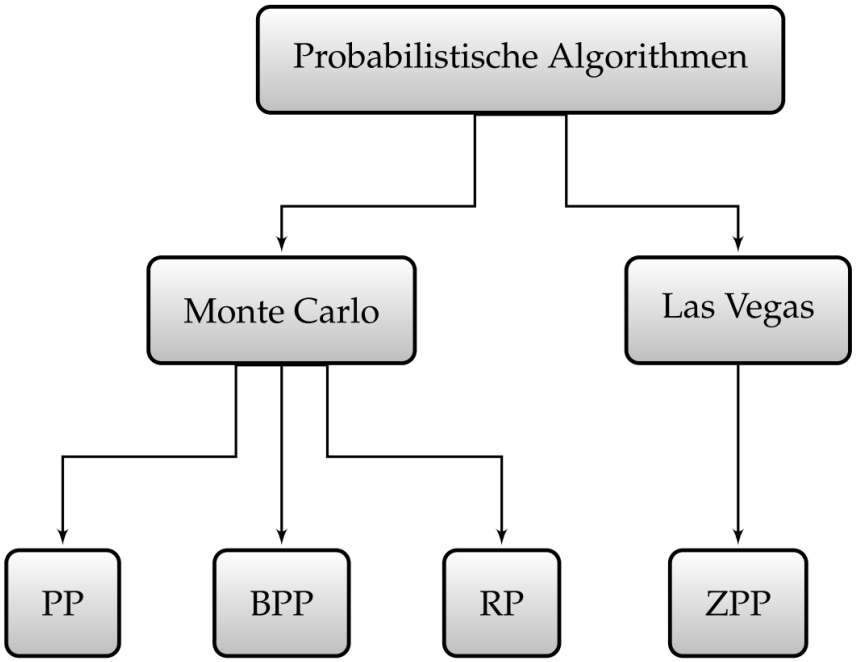
- Ziel: Optimierung linearer Zielfunktionen über einer Menge, die durch lineare (Un-)Gleichungen eingeschränkt ist
- Simplexverfahren
- Ellipsoid-Methode

12. A*-Suchalgorithmus



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

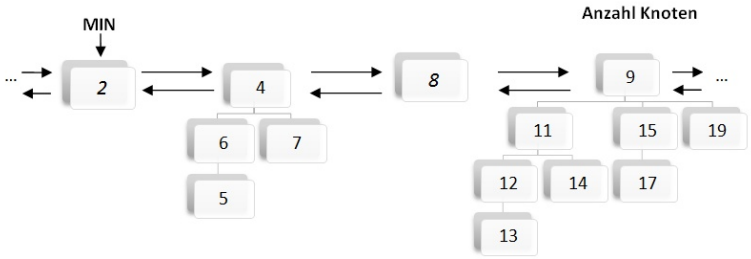
13. Randomisierte Algorithmen



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

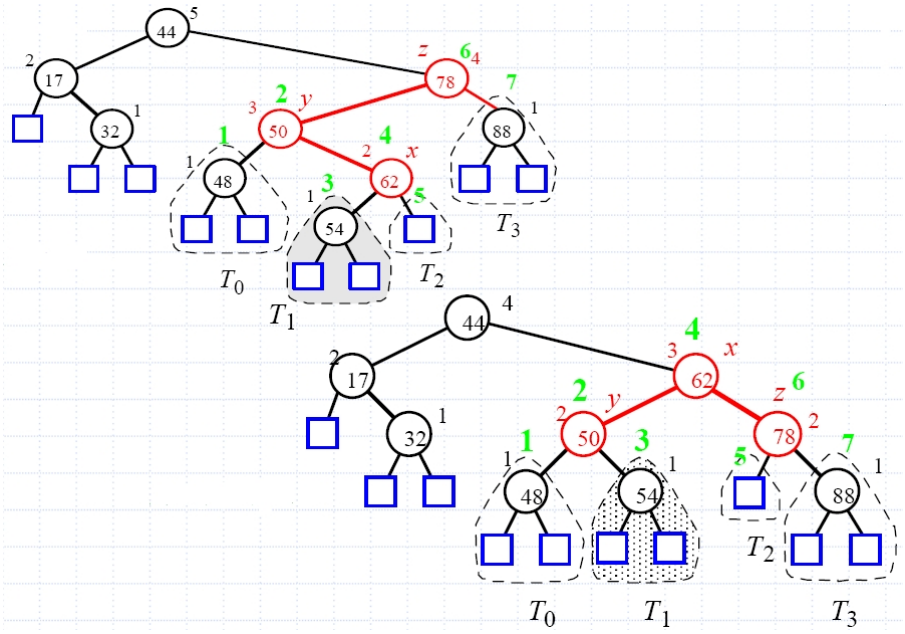
Übersicht

14. Fibonacci-Heaps



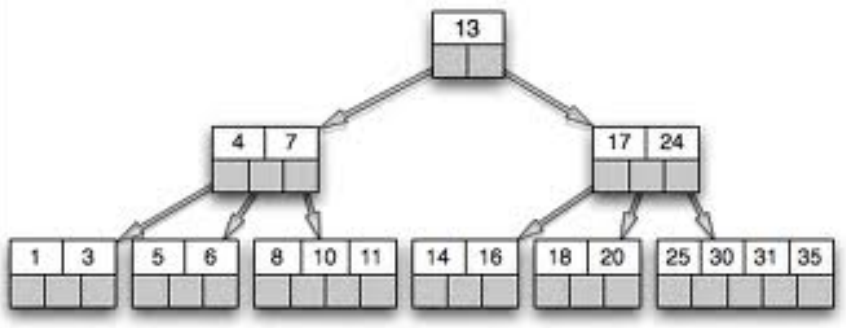
- Datenstruktur zur Implementierung einer Vorrangwarteschlange (*priority queue*)
- Liste von geordneten Bäumen
- Heap-Bedingung: Priorität jedes Knotens mindestens so groß wie Priorität seiner Kinder
- (Fast) alle Operationen haben amortisiert konstante Laufzeit

15. AVL-Bäume



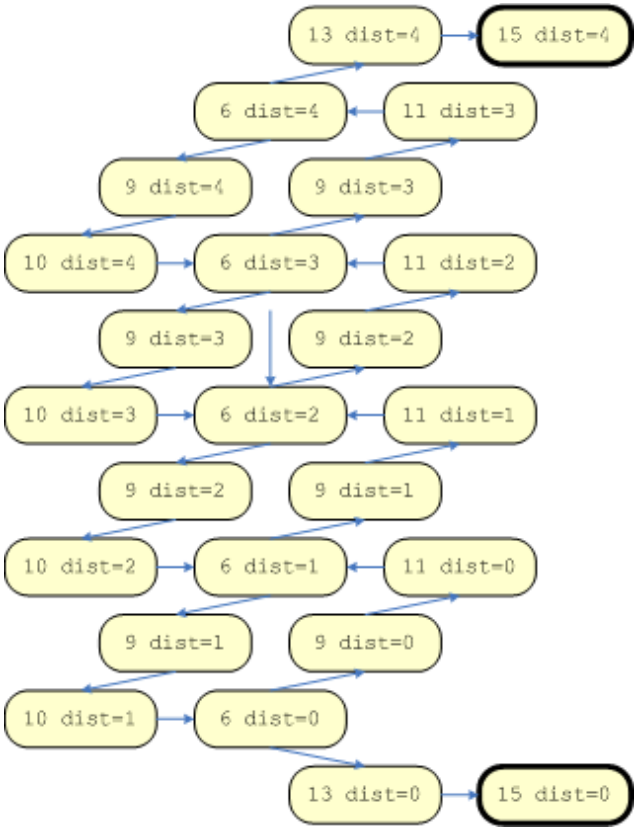
- Balancierter binärer Suchbaum
- Invariante: maximaler Höhenunterschied der Teilbäume jedes Knotens ist 1
- Worst-Case-Komplexität der üblichen Operationen: $O(\log n)$

16. B-Bäume



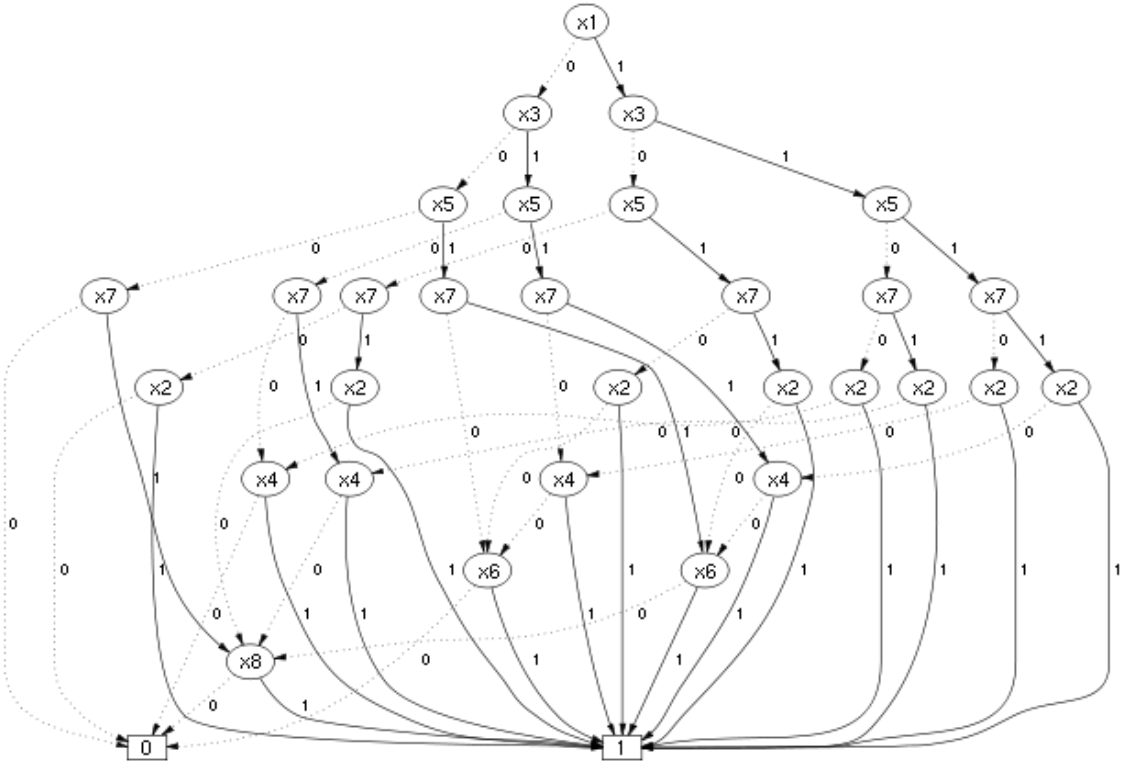
- Immer vollständig balanciert
- Besonderheit: Anzahl der Knotennachfolger variabel (mit Obergrenze)
- Einsatz vor allem in Datenbanken und Dateisystemen

17. Bitstate Hashing



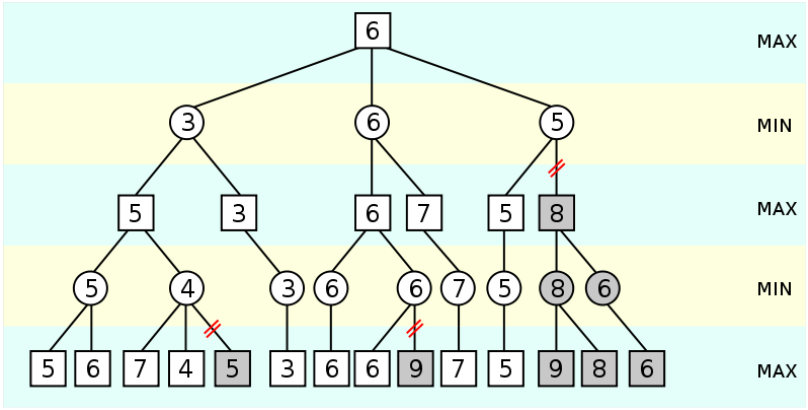
- Effiziente Darstellung von Zustandsräumen
- Problem: Erkennung von Zykeln
- Idee: Benutzung einer Hashfunktion, Speicherung von 0/1 im Hasharray
- Bis zu 98% Speicherersparnis im SPIN-Tool

18. Binäre Entscheidungsdiagramme



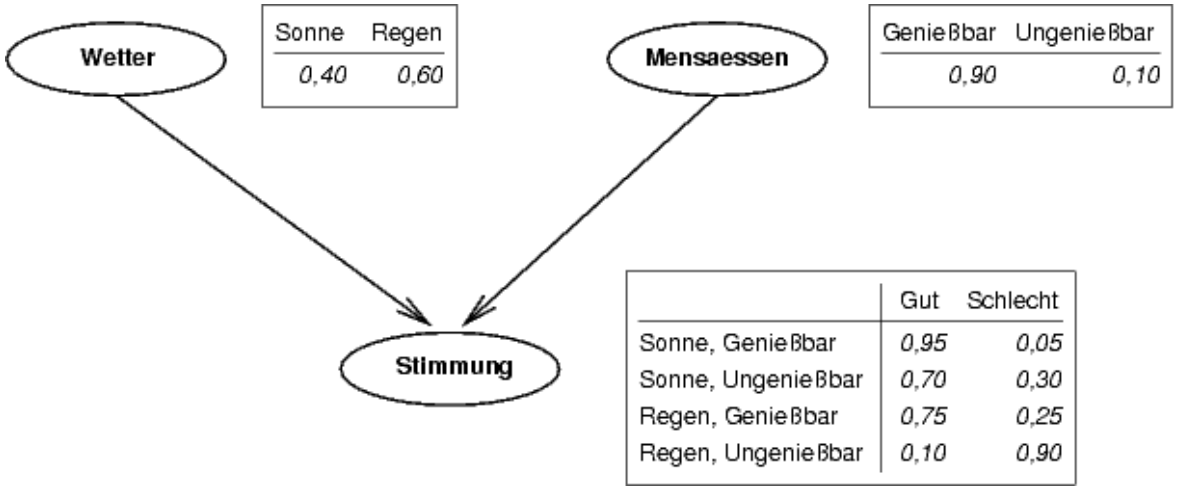
- Datenstruktur zur Darstellung Boolescher Funktionen über mehreren Variablen
- Einsatz: Verifikation von (Hardware-)Systemen
- Effizient durch Sharing gemeinsamer Teilbäume
- Problem: Finden geeigneter Variablenordnungen

19. 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

20. Bayessche Netze



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