

Kapitel 2

Datenstrukturen

2.1 Situation

(Jim Gray, 97)



universität
wien

Datenstrukturen (Datenbanken) speichern ALLE Daten

The New World:

Milliarden von Objekten

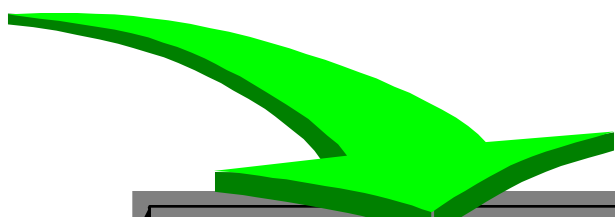
Große Objekte (1MB)










The Old World:

Millionen von Objekten

100-Byte Objekte

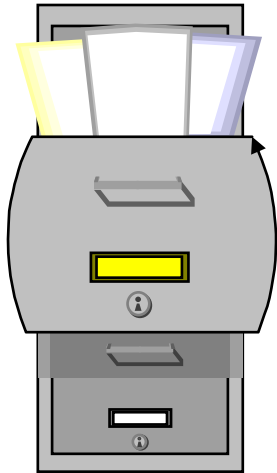
People	
Name	Address
David	NY
Mike	Berk
Won	Austin



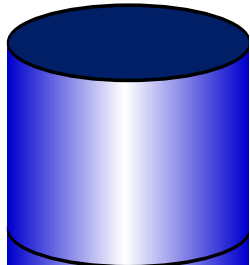
People				
Name	Address	Papers	Picture	Voice
David	NY			
Mike	Berk			
Won	Austin			

Paperless office
Library of congress online
All information online
entertainment
publishing
business
Information Network,
Knowledge Navigator,
Information at your fingertips

Magnetische Platten sind billiger als Papier *(Jim Gray, 97)*



Dateischrank:	cabinet (4 drawer)	250\$
	paper (24,000 sheets)	250\$
	space (2x3 @ 10\$/ft ²)	180\$
	total	700\$
		3 ¢/sheet



Platte:	disk (4 GB =)	500\$
	ASCII: 2 m pages	
	(100x cheaper)	0.025 ¢/sheet

Image:	200 k pages	
	(10x cheaper)	.25 ¢/sheet

Conclusio: Speichere alles auf Platten

XXX verdoppelt sich alle 18 (24) Monate

60% Steigerung pro Jahr

Micro Prozessor Geschwindigkeit

Chip Dichte

Magnetische Platten Dichte

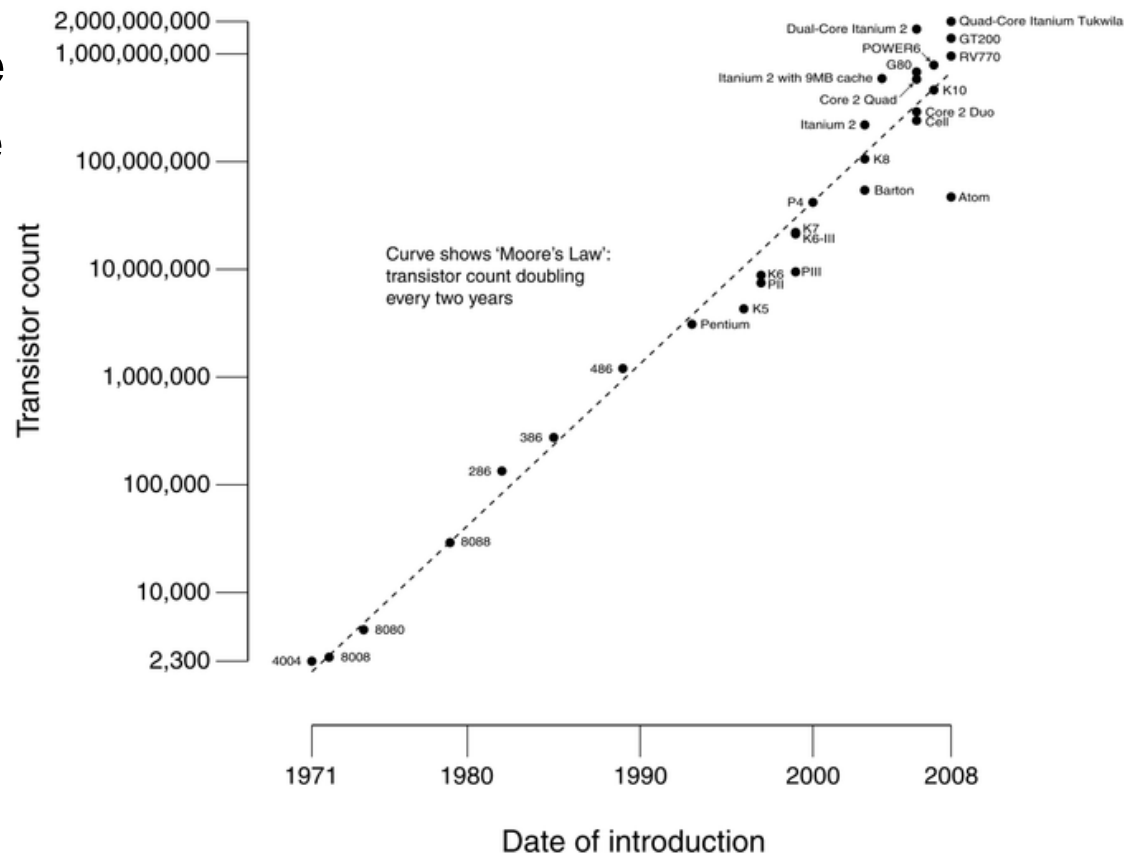
Kommunikationsbandbreite

WAN Bandbreite

nähert sich LAN

Plattenzugriffs-
geschwindigkeit ???

CPU Transistor Counts 1971-2008 & Moore's Law



Magellan Projekt

Satellit umkreiste die Venus, Radarabtastung
zur Oberflächendarstellung

Sandte 3 Terabyte Daten

Rendering der Daten benötigt 13 Gigabyte / sec
War damals (1994) technisch nicht durchführbar!

<http://www2.jpl.nasa.gov/magellan/>



Wettervorhersage

Zirkulationsmodelle der Atmosphäre und der Ozeane

1000 Jahre Simulation, 150 km² Auflösung,
0.2 simulierte Jahre / Maschinenstunde

Ein Durchlauf auf Intel Touchstone Delta 57 Wochen

40 MB Daten / Simulationsminute = 20 Terabytes



CERNs Herausforderung: Datagrid

Neuer Beschleuniger LHC mit 4 Detektoren

Large Hadron Collider, 14 TeV

Ziele: Suche nach Higgs Boson
und Graviton (et al.)

Start 2009

Ziele

Weltweiter Zugriff auf die Daten

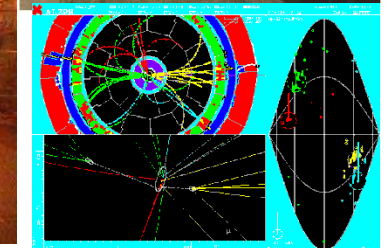
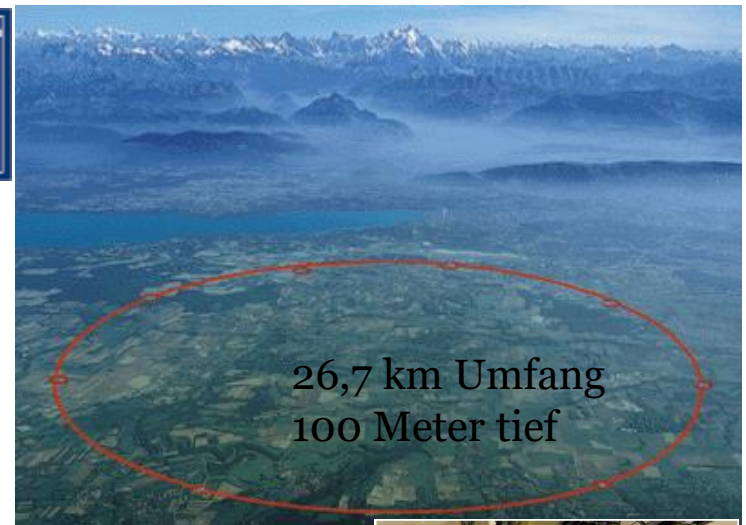
CERN und Regional Centers (Europa, Asien, Amerika)

2000 Benutzer

Riesige Datenvolumen

Daten Semantik

Performance und throughput



Charakteristische Größen

1-6 (vielleicht 100 ?) Petabyte / Jahr

Zeitraum 15 bis 20 Jahre

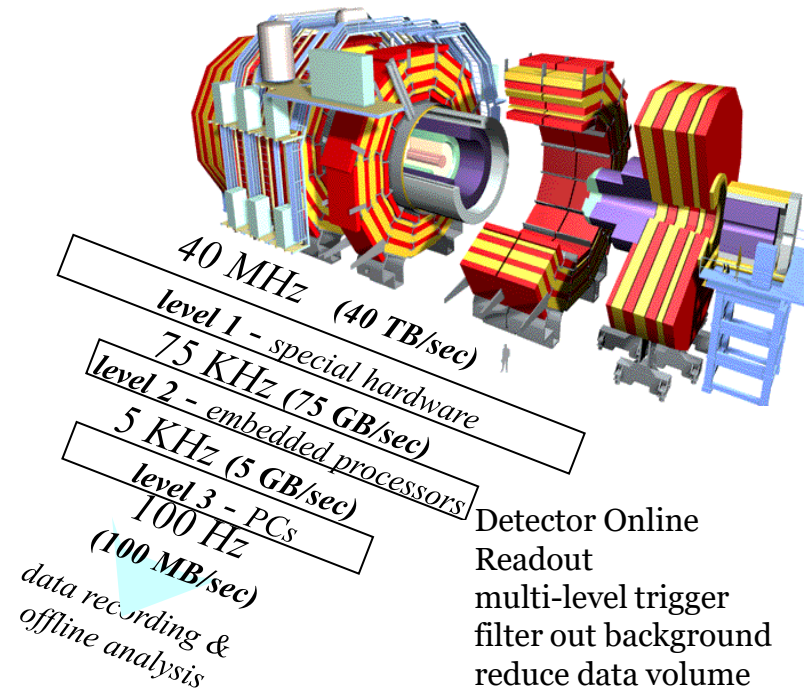
CERN Tier 0 / Weltweit (2009)

24000 / 61500 CPUs

5.6 / 35 PB Platten Platz

22 / 40 PB Bänder

340 Gigabyte IO Bandbreite



Größe: Was ist ein Petabyte?

1 Petabyte = 2 hoch 50, i.e.
(1,125,899,906,842,624) bytes $\sim 10^{15}$ bytes

1,000,000,000,000 business letters
100,000,000,000 book pages
50,000,000,000 FAX images
10,000,000,000 TV pictures (mpeg)
4,000,000 LandSat images

150,000 miles of bookshelf
15,000 miles of bookshelf
7,000 miles of bookshelf
10,000 days of video

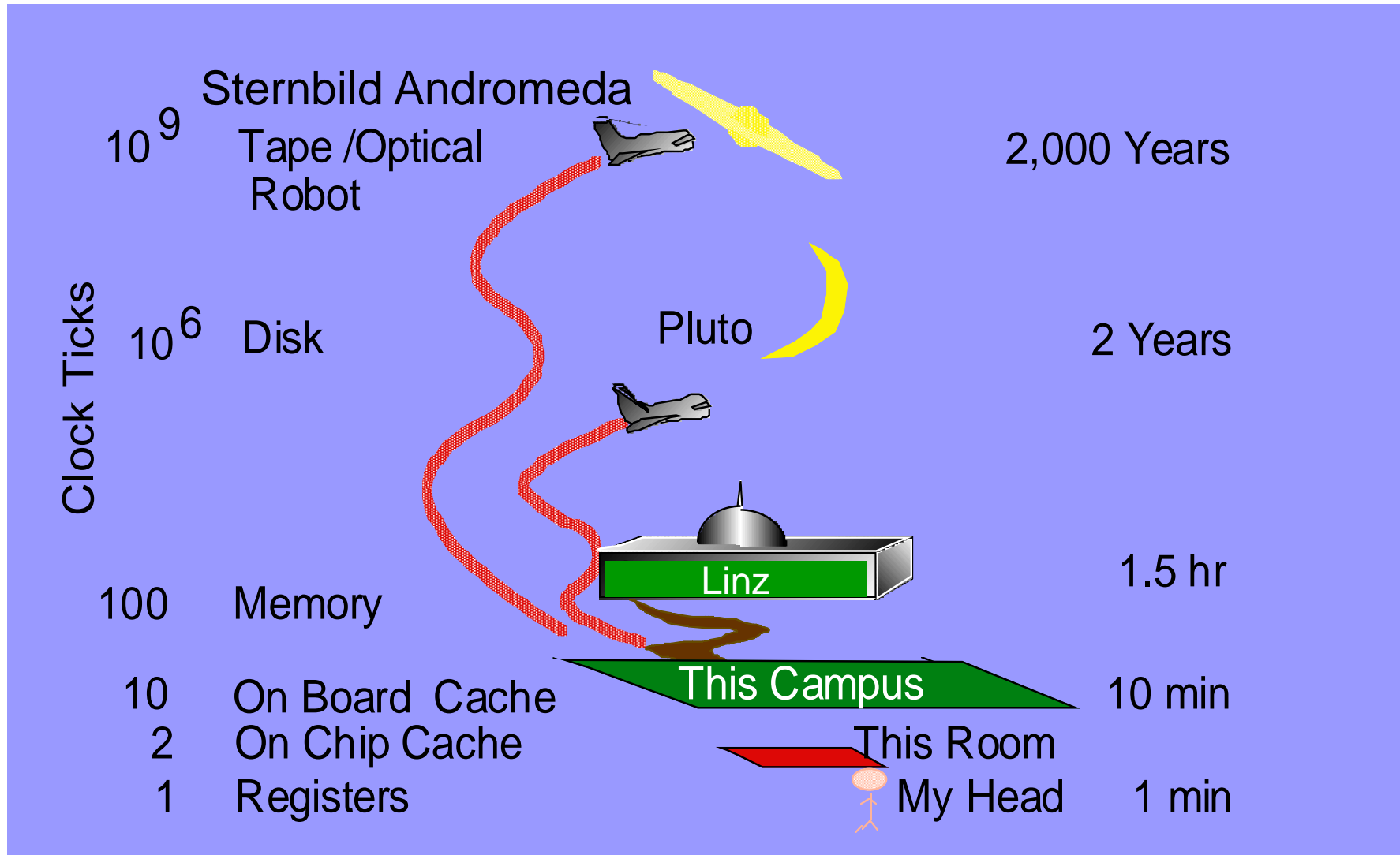
Library of Congress (in ASCII)
enthält 0.025 Petabyte



Aktuelle und zukünftige Projekte generieren weit mehr Daten
Auf uns warten Exa-, Zeta-, Yotta Byte !!!

Geschwindigkeit: Speicherzugriffszeiten

Wie weit sind die Daten entfernt ? (Jim Gray, 97)



Beispiel: 100 Telefonnummern zu verwalten

Ein „Haufen“ Zettel mit Namen und Nummern

Finden einer Telefonnummern durch sequentielle Suche
benötigt im Durchschnitt 50 „Zugriffe“

Zettel nach dem Namen sortiert

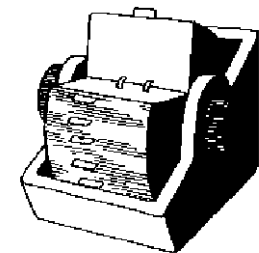
Suche durch binäres Aufteilen („in die Mitte, Schlüssel-vergleich und dann
links oder rechts davon weitersuchen“)

Ungefähr $\lg 100 \approx 7$ Zugriffe

Rolodex

Zettel sortiert, in Ordnern und mit Namensindex

Ziel mit einem (1!) Zugriff gewünschte Nummer



Information “effizient” zu verwalten!

Was bedeutet “effizient”?

Quantitative Ziele

Zugriffszeit

schnelles Einfügen, Verändern, Löschen, ... (d.i. “Bearbeiten” im weitesten Sinn)
der Daten

Speicherplatz

kompaktes Speichern der Information

Qualitative Ziele

Unterstützung spezifischer Zugriffsarten auf Eigenschaften bzw.
Charakteristiken der Daten



Erfüllung dieser Ziele führte zur Entwicklung von

Datenstrukturen

Datenstrukturen dienen zur Verwaltung großer Mengen ähnlicher Objekte

Unterschiedliche Datenstrukturen dienen zur Verwaltung unterschiedlicher Objekte, die durch unterschiedliche Eigenschaften charakterisiert sind, daraus folgt:

Für unterschiedliche Problemstellungen unterschiedliche Datenstrukturen!

Suche in einem Telefonbuch mit 2000000 Einträgen

Annahme: Zugriff auf einen Datensatz in 0,01 ms

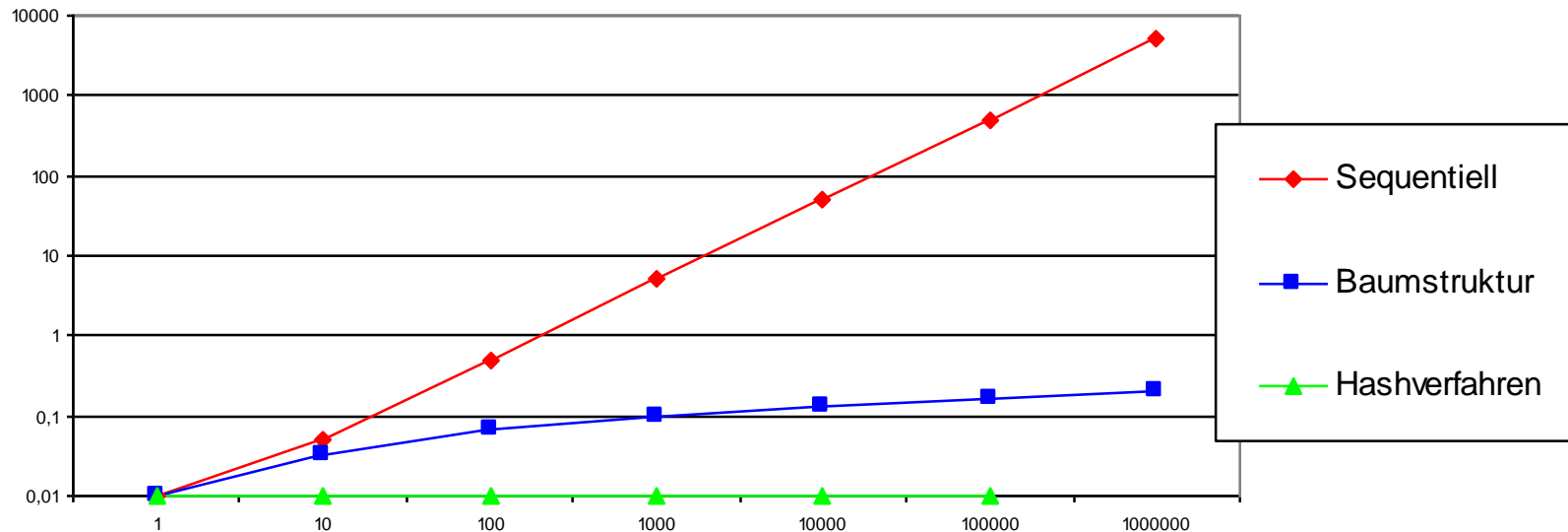
Ansätze (Zeit für einen Zugriff)

Sequentielles Suchen (im Mittel $1000000 * 0,01 \text{ ms} = 10 \text{ s}$)

Baumstruktur (ungefähr $\lg 2000000 * 0,01 = 0,21 \text{ ms}$)

Hashverwaltung (1 Zugriff = 0,01 ms)

Zugriffzeit im Verhältnis zur Dateigröße



Alle bekannten Datenorganisationsformen bauen auf einigen wenigen einfachen Techniken auf

Sequentielle Techniken

Listen

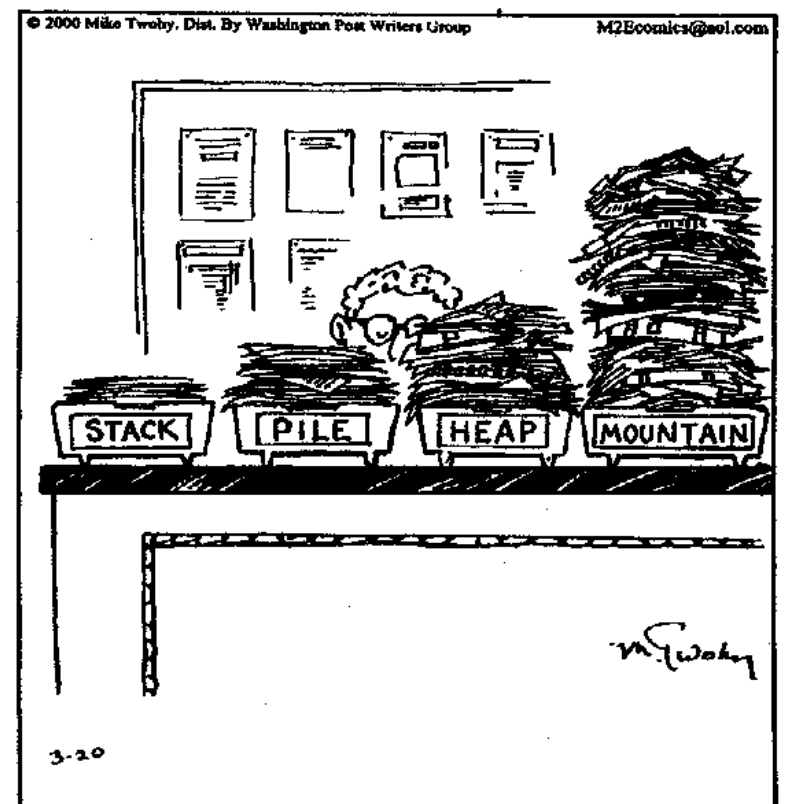
Stack, Queue

Hashverfahren

Dictionary, Hash Tabelle,
Kollisionsverfahren

Baumstrukturen

Binärer Baum, B+-Baum,
Priority Queue



Ein Vektor (vector, Feld) verwaltet eine fix vorgegebene Anzahl von Elementen eines einheitlichen Typs.

Zugriff auf ein Element über einen ganzzahligen Index (die Position im Vektor)

Aufwand des Zugriffes für alle Elemente konstant



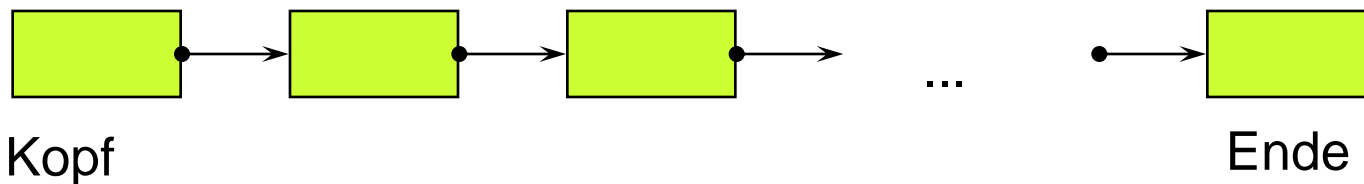
Anwendungen

Verwaltung fix vorgegebener Anreihungen, Strings, math. Konzepte

Eine Liste (list) dient zur Verwaltung einer beliebig Anzahl von Elementen eines einheitlichen Typs.

Die Elemente werden in einer Sequenz angeordnet, die sich (meist) aus der Eintragereihenfolge ableiten lässt (ungeordnet).

Der Aufwand des Zugriffs auf ein einzelnes Element ist abhängig von der Position in der Sequenz.



Anwendungen

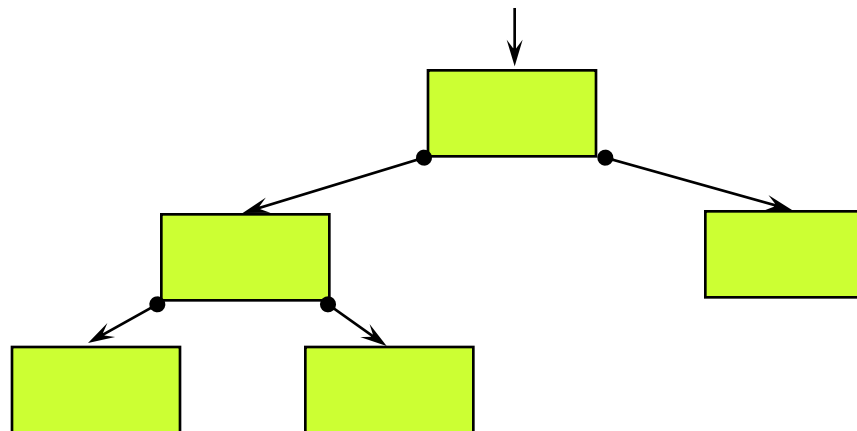
sequentielle Datenbestände, große Datenmengen, externe
Speicherung

Der Baum (tree) stellt eine Generalisierung der Liste auf eine 2-dimensionale Datenstruktur dar.

besteht aus Knoten und Kanten

Exponentieller Zusammenhang zwischen Tiefe des Baumes und Anzahl der Knoten

Anwendungen: allgemeine Schlüsselerwaltung, Haupt- und Externspeichermanagement



Dynamik

- Datenverwaltung

 - Einfügen, Löschen

- Datenmenge

 - beliebige oder fixe Anzahl von Elementen

Aufwand

- Laufzeit der Operationen

- Speicherplatzverbrauch

Modell

- Operationenumfang

Datenstruktur	Stärken	Schwächen
Vektor	dynamische Verwaltung direkter Elementzugriff konstanter Aufwand der Operationen geringer Speicherplatz	oft statisch (nur beschränkte Datenmenge) eingeschränkte Operationen
Liste	dynamische Verwaltung beliebige Datenmenge klares Modell	linearer Aufwand der Operationen simples Modell
Baum	meist dynamische Verwaltung beliebige Datenmenge logarithmischer Aufwand der Operationen	Balanzierungsalgorithmen relativ hoher Speicherplatzverbrauch komplexes Modell manchmal nur Einfüge-Operation unterstützt

Datenorganisation

- Effizienz

- Quantität - Qualität

Datenstrukturen

- Typen

- Vergleichskriterien