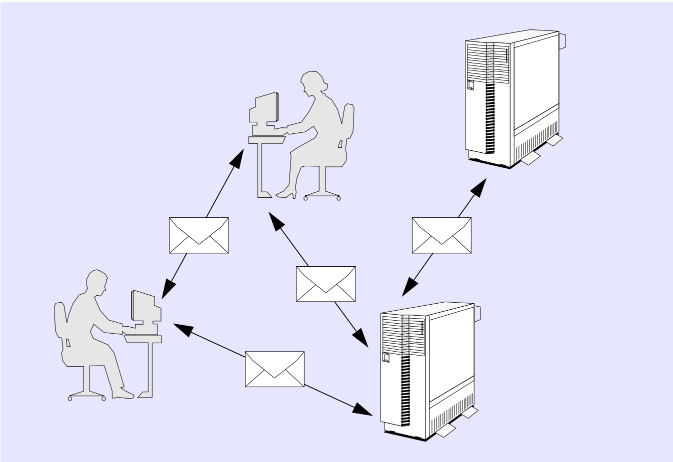
Verteilte Systeme

Friedemann Mattern

Departement Informatik
ETH Zürich



- Rechner, Personen, Prozesse, “Agenten” sind an *verschiedenen Orten*.
- Autonome Handlungsträger, die jedoch gelegentlich *kooperieren* (und dazu über Nachrichten *kommunizieren*).

Übersicht I

1. Verteilte Systeme: Definition und Motivation
2. Multiprozessoren, Multicomputer
3. Überblick Rechnernetze
   - LAN, WAN
   - Schichtenmodell, Protokolle

   teilweise Wiederholung aus dem Grundstudium
4. Kommunikation
   - Mitteilung / Auftrag
   - synchron / asynchron
   - Puffer
   - RPC (Protokolle, Fehlersemantik)
   - Mailbox, Ports, Kanäle
   - Gruppenkommunikation (Broadcast-Semantik)
   - verteilte Ereignisse
   - Tupelräume
5. Verteilte Programmiersprachen
   - Occam als Beispiel
6. Namensverwaltung
7. Schutz und Sicherheit
   - Autorisierung, Capabilities
   - Authentifizierung
   - DES, Public-key-Systeme
   - Kerberos

   teilweise Wiederholung aus dem Grundstudium

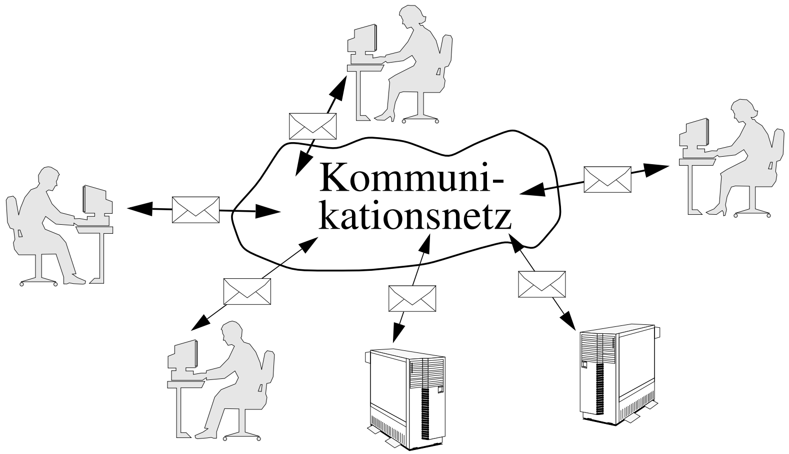
Übersicht II

8. Client / Server-Modell
   - verteilte Betriebssysteme
   - konkurrente Server und Servergruppen
   - zustandsbehaftete / zustandslose Server
   - verteilte Dienste
9. Middleware: Software für offene, verteilte Systeme
   - Sun-RPC
   - OSF-DCE
   - CORBA
   - Jini
10. Logische Zeit, verteilte Algorithmen
    - Lamport-Uhren
    - wechselseitiger Ausschluss
11. Physische Zeit, Uhrensynchronisation
12. Mobiler Code und mobile Agenten

"Verteiltes System" - zwei Definitionen

A distributed computing system consists of multiple autonomous processors that do not share primary memory, but cooperate by sending messages over a communication network.

-- *H. Bal*



A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.

-- *Leslie Lamport*

- welche Problemaspekte stecken hinter Lamports Charakterisierung?

Organisatorisches zur Vorlesung

4-stündige Vorlesung (inkl. Übungen)

Sinnvolle Vorkenntnisse:

- Betriebssysteme (Prozessbegriff, Synchronisation)...
- ggf. UNIX / C / Java
- Grundkenntnisse der Informatik und Mathematik (Vordiplom)

	Mo	Di	Mi	Do	Fr
08.00 – 10.00		**Vert Sys**			
10.00 – 11.30		IFW A36			
11.30 – 13.30					
13.30 – 15.00					IFW A32
15.00 – 17.00					**Vert Sys**

Di 08:15 - 10:00, IFW A36
Fr 15:15 - 17:00, IFW A32
} *Vorlesung*

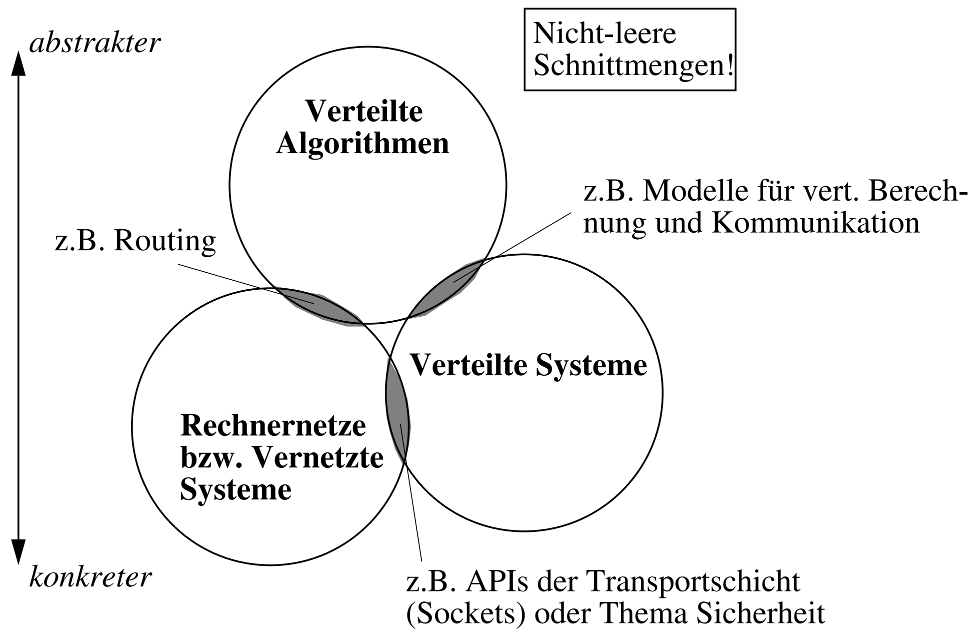
Absicht!

- Gelegentliche *Denkaufgaben* in der Vorlesung
- Praktische Übungen in Form eines begleitenden *Praktikums* (korreliert nur schwach mit dem Inhalt der Vorlesung) *komplementieren* Vorlesung
- Gelegentliche *Übungsstunden* (zu den Vorlesungsterminen) zur Besprechung des Praktikumteils

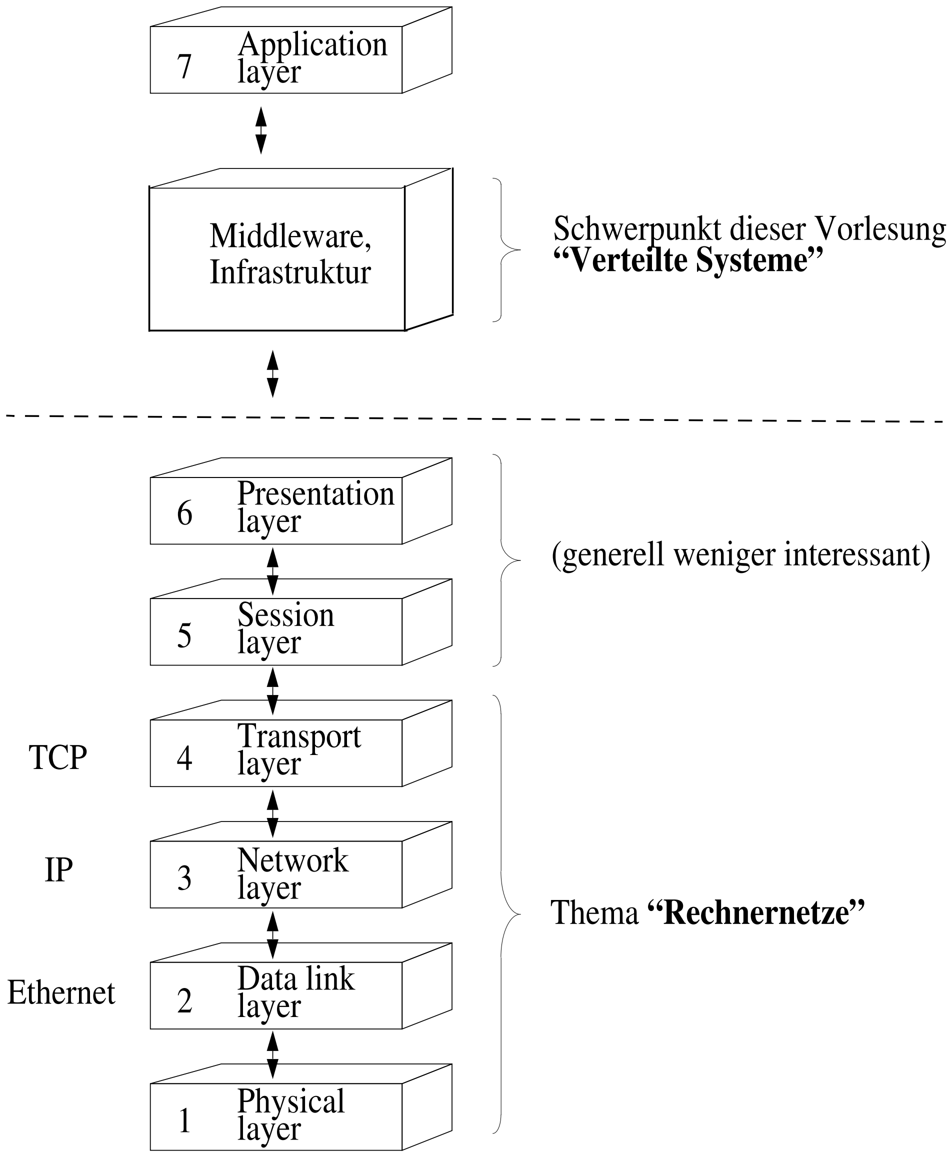
- *Folienkopien* jeweils in der darauffolgenden Vorlesung?
- auch im WWW im .ps- und .pdf-Format: www.inf.ethz.ch/vs/edu

Thematisch verwandte Veranstaltungen im Fachstudium

- Mattern (bzw. Plattner): *Verteilte Systeme*, 4st
- Mattern: *Verteilte Algorithmen*, 3st (WS 01/02)
- *Ubiquitous Computing* (SS 02)
- *Mobile Computing* (geplant)
- Einschlägige *Seminare*
- *Semester- und Diplomarbeiten*



Netze, Anwendungen, Verteilte Systeme



Literatur (vorläufig)

G. Coulouris, J. Dollimore, T. Kindberg: *Distributed Systems: Concepts and Design (3rd ed.).* Addison-Wesley, 2001

M. Weber: *Verteilte Systeme*. Spektrum Hochschultaschenbuch, 1998

S. Mullender (Hg.): *Distributed Systems (2nd ed).* ACM Press and Addison-Wesley, 1994

A. Tanenbaum: *Distributed Operating Systems*. Prentice-Hall, 1995

R.G. Herrtwich, G. Hommel: *Nebenläufige Programme*. Springer-Verlag, 1994

K. Geihs: *Client/Server-Systeme*. Internat. Thomson Publ., 1995

A. Schill: *DCE - Das OSF Distributed Computing Environment*. Springer-Verlag, 1993

W.K. Edwards: *Core Jini*. Prentice Hall, 1999

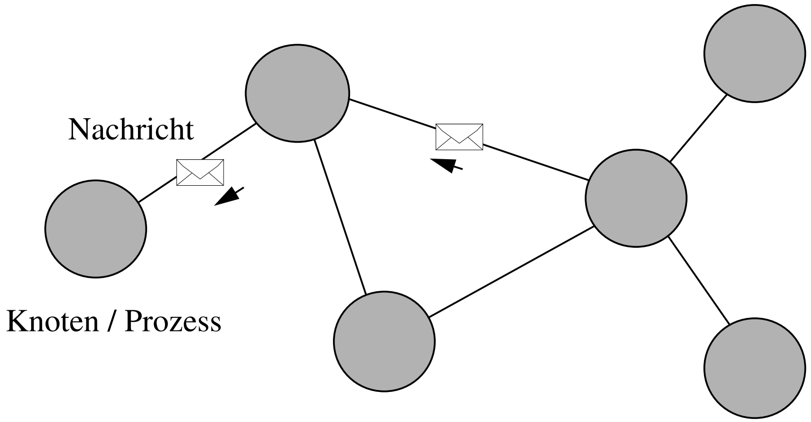
A. Tanenbaum: *Computer Networks (3rd ed.).* Prentice-Hall, 1996

D. Comer: *Internetworking with TCP/IP, Volume III, Client-Server Programming*. Prentice-Hall, 1993

B. Schneier: *Applied Cryptography (2nd ed.).* Wiley, 1996

In den aufgeführten Büchern findet man weitere Literaturangaben.

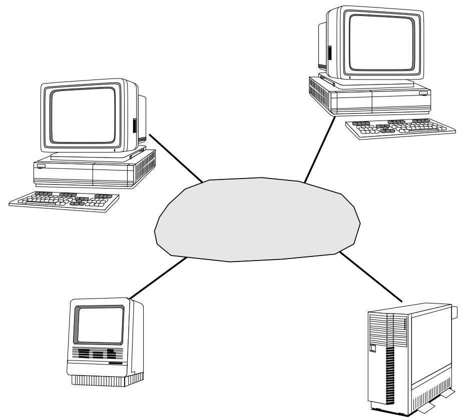
“Verteiltes System”



Physisch verteiltes System:
Mehrrechnersystem ... Rechnernetze

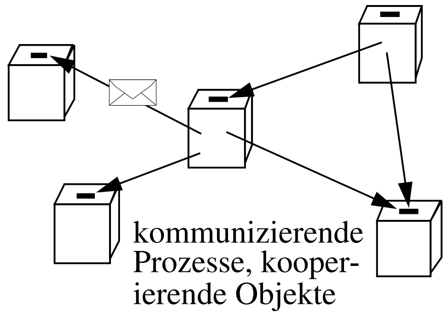
Logisch verteiltes System: Prozesse (Objekte, Agenten)
- Verteilung des Zustandes (keine globale Sicht)
- Keine gemeinsame Zeit (globale, genaue "Uhr")

Sichten verteilter Systeme



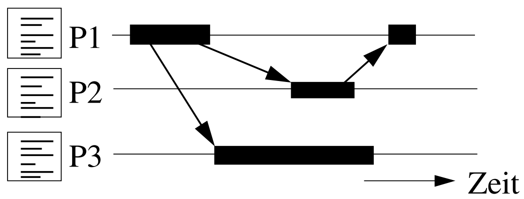
Rechnernetz mit Rechenknoten:
- Multicomputer (Parallelrechner)
- LAN = Local Area Network
- WAN = Wide Area Network
- Routing, Adressierung....

zunehmende Abstraktion



Objekte eines *Betriebssystems* bzw. einer *Programmiersprache*

==> “Programmierersicht” (Client, Server...)



Algorithmen- und Protokollebene

- Aktionen, Ereignisfolgen
- Konsistenz, Korrektheit

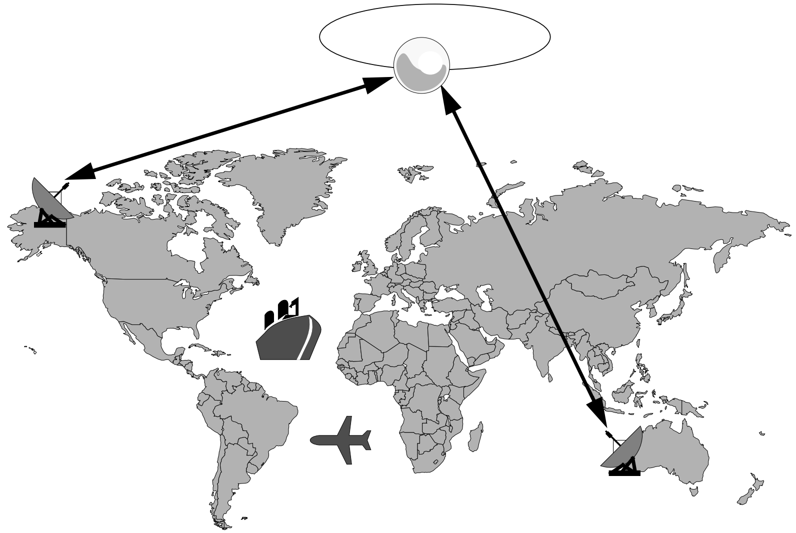
Sichten (2)

- Von Technologie bis zu Prinzipien
  - physische Sicht: z.B. Techniker
  - Programmiersicht: z.B. Programmierer, Systementwickler
  - abstrakte Sicht: z.B. Algorithmentheoretiker

- Entspricht dem Kanon der verschiedenen Lehrveranstaltungen:
  - „Rechnernetze“: technische Eigenschaften, Topologien, Realisierungsweisen
  - **„Verteilte Systeme“**: Infrastruktur, Komponenten, Protokolle
  - „Verteilte Algorithmen“: Korrektheit, grundsätzliche Phänomene, Konzeptualisierung

- „*Logisch* verteilte Systeme“: unabhängig von *physischer* Verteilung
  - fehlender globaler Zustand, fehlende globale Zeit
  - Parallelität
  - Autonomie (z.B. unabhängige Prozesse, Objekte, Agenten …)

Die verteilte Welt



Auch die "reale Welt" ist ein verteiltes System:

- Viele gleichzeitige (“parallele”) Aktivitäten
- Exakte globale Zeit nicht erfahrbar / vorhanden
- Keine konsistente Sicht des Gesamtzustandes
- Kooperation durch explizite Kommunikation
- *Ursache* und *Wirkung* zeitlich (und räumlich) getrennt

- “Inkonsistente Zustände”: Kriegsende zwischen England und Frankreich war in den Kolonialgebieten erst später bekannt!
- Heute: “zeitkompakter Globus” - weitgehend synchronisierte Uhren.

Motivation (1)

- Was sind sinnvolle verteilte Anwendungen?

- Worin besteht der letztendliche Nutzen?

Ein Szenario:

Die weltweit verteilte virtuelle Bibliothek

Notwendig:

- schnellere Kommunikationsnetze
- bessere Geräte (sehr hohe Auflösung, flach, portabel...)
- elektronische Speicherung (fast) aller Bücher, Dokumente...
- Software-Infrastruktur, Standards...
- soziale, politische, ökonomische "Akzeptanz"

schwierig

am schwierigsten

Vorteile:

- schnelle Verfügbarkeit, neueste Version
- Kostensenkung
- Suchfunktionen, elektronische Auswertbarkeit
- Querbezüge durch explizite oder implizite Referenzen (--> Hyperlinks)
- Integration verschiedener Medien: Text, Sprache, Bilder, Video, Animationen... --> Multimedia

Konsequenzen?

(sozial, rechtlich, kulturell, psychisch, ökonomisch...)

Einige Bemerkungen dazu

- Abrechnung fälschungssicher
- Copyright garantieren
- Authentizität garantieren
- Vertraulichkeit gewährleisten (was liest Mr. X?)
- Ausfallsicherheit
- Suchsystem (Indizes, Metadaten, Suchmaschinen...)
- Ortstransparenz
- Heterogenität (Geräte, Standards,...)
- Effizienz (z.B. ggf. parallele Suche im Netz)
- Dezentrale Organisation
- ...

- Architekturaspekte
    - Systemarchitektur soll u.a. obige Aspekte berücksichtigen
    - Schnittstellen, Teilsysteme
    - "offen" zu existierenden Diensten, anderen Systemen
    - viele Entwurfsentscheidungen (z.B.: data shipping vs. function shipping bei Suchfunktionen)

Motivation (2): Bessere Nutzung global verfügbarer Ressourcen

- WWW-Zugriff auf weltweite Wissensbestände

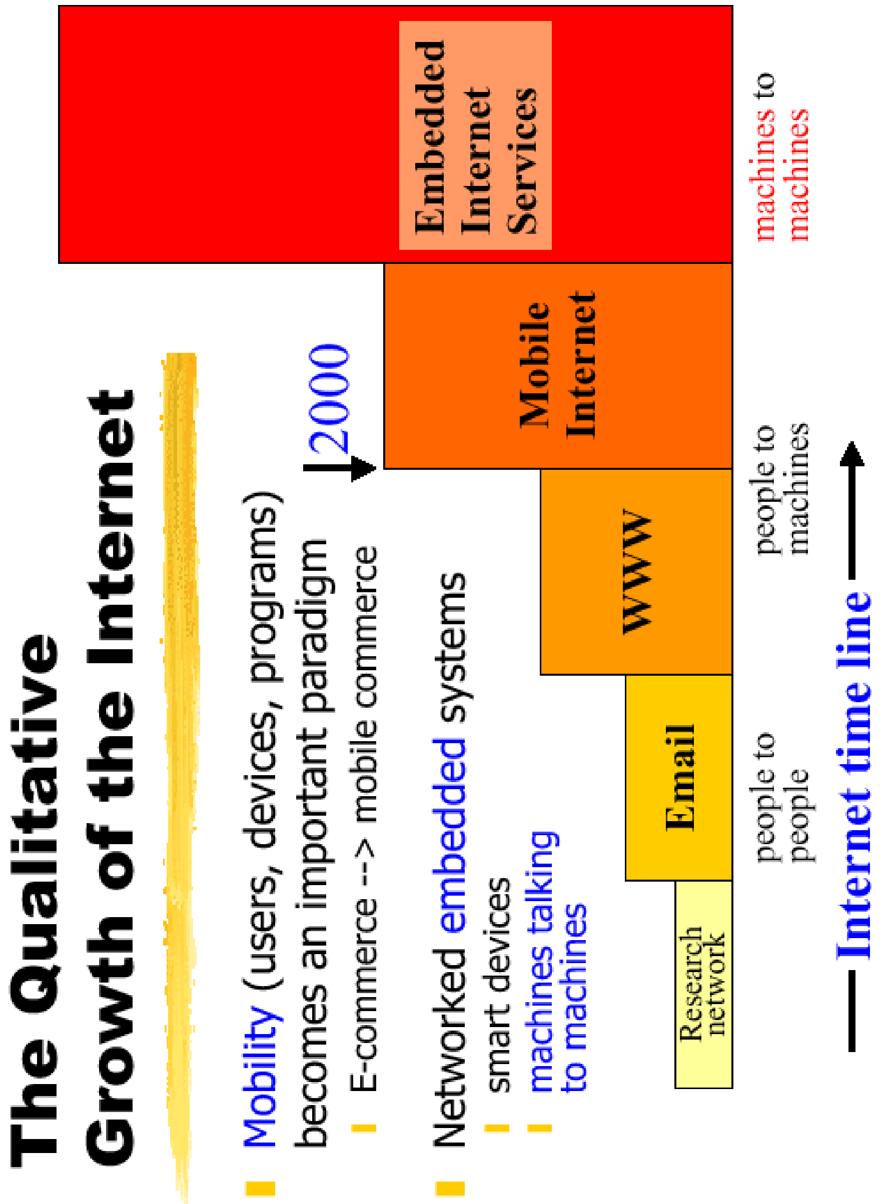
- Nutzung von Hochleistungsrechnern oder Spezialsystemen
    - Supercomputer
    - Software-Bibliotheken

- „Aufsammeln“ ungenutzter Rechenleistung im Internet
    - Faktorisierung grosser Zahlen per Email
    - Brechen einer DES-Verschlüsselung

- Nutzung der Zeitverschiebung zwischen Kontinenten
    - billige „Nachtrechenzeit“ für Anwender mit normaler Tagesarbeitszeit
    - Hotline rund um die Uhr

- Bearbeitung eines Problems an mehreren Standorten
    - internationaler Konzern mit nationalen Niederlassungen
    - internationale Spezialistenteams (Medizin, Forschung)

===> ökonomischer Effekt
+ ermöglicht “Globalisierung” } auf der Basis der Vernetzung (“Internet”)



Motivation (3): Middleware für das Internet

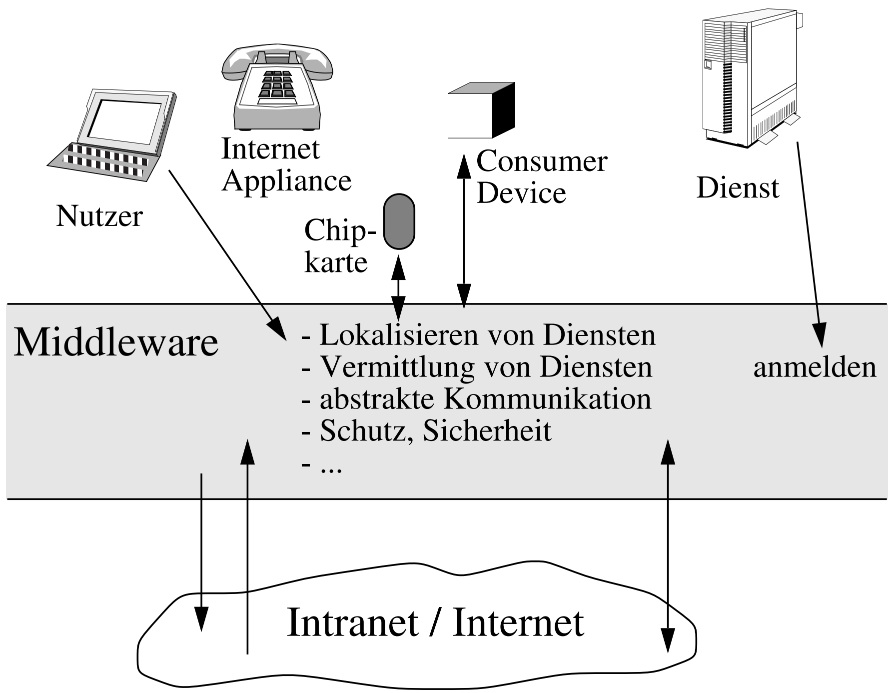
- Phänomen: das Internet verbreitet sich immer weiter
    - mehr Nutzer, Popularisierung
    - bis in die Haushalte
    - immer exotischere Endgeräte (PDA, Handy, Kühlschrank, Chipkarte)
    - bald enthalten vielleicht auch Briefmarken, Kleidungsstücke etc. kommunikationsfähige Chips
- Mobile Geräte, dynamische Umgebungen
- Es entstehen neue Dienste im Netz
- Dienste interagieren miteinander
    - Kompatibilität, Standards, Protokolle, offene Schnittstellen...
- Markt erfordert sehr schnelle Reaktion
    - schnelle Implementierung neuer Dienste
    - Update über das Netz
- Anschluss neuer Geräte muss “von selbst” erfolgen
    - Integration in eine Infrastruktur und Umgebung von Ressourcen

---

- Kann man eine Infrastruktur schaffen, die das unterstützt?
    - wichtig auch für Electronic Commerce-Szenarien

Middleware für das Internet

- Plattform / Infrastruktur für verteilte Anwendungen



- Beispiel: *Jini*
    - Zweck: Interaktion mit dem Netz und mit Diensten vereinfachen
    - Lookup-Service (“Bulletin-Board”)
    - “Leasing” von Objekten (Freigabe bei Ablauf des Vertrages)
    - hot plugging von Objekten, Teildiensten etc.
    - Garbage-Collection von Objekten im Netz
    - Vermittlung von Ereignissen (events); API für events
    - Unterstützung mobiler Geräte und Dienste
    - Föderation kooperierender Java-VMs (Gruppenkonzept)
    - Mobiler Code (Java-Bytecode, Applet); z.B. Druckertreiber als “Proxy”
    - Kommunikation über entfernter Methodenaufruf oder (persistente) Tupel-Räume

Transparenz

Transparenz = unsichtbar ("durchsichtig") sein

Verteiltheit ("Separation") wird vor dem Benutzer / Anwendungsprogrammierer verborgen, so dass das System als Ganzes gesehen wird (statt als Menge von Einzelkomponenten)

--> Umgang mit der Verteiltheit wird einfacher
--> Abstraktion von "internen" Aspekten

Verschiedene Arten der Transparenz, z.B.:

Ortstransparenz

Ort, an dem sich Daten befinden oder an dem ein Programm ausgeführt wird, ist unsichtbar

Replikationstransparenz

Unsichtbar, wieviele Replikate eines Objektes (z.B. Datei) existieren

Concurrency-Transparenz

Mehrere Benutzer / Prozesse können gemeinsame Objekte (z.B. Dateien) benutzen, ohne dass es zu Inkonsistenzen kommt

Leistungstransparenz

Kein (spürbarer) Leistungsunterschied zwischen lokaler und entfernter Bearbeitung

Ausfalltransparenz

Ausfall einzelner Komponenten ist unsichtbar --> Fehlertoleranz

Transparenz (2)

Aufwand zur Realisierung von Transparenz ist hoch!

Implementierung von Transparenz auf verschiedenen Ebenen möglich:

- Betriebssystem (--> alle Anwendungen profitieren davon)
- Subkomponenten (z.B. Dateisystem)
- Anwendungsprogramm (Nutzung der Semantik)

Transparenz ist *graduelle Eigenschaft.*

Bsp. Ortstransparenz:

schwach: Benutzer sieht verschiedene Rechner

- Zielrechner muss bei Prozessgründung, login... angegeben werden
- Bsp. UNIX: remote login, ftp, rexec

mittel: Einige einfache "verteilte Anwendungen"

- Bsp. UNIX: rwho, email (ortstransparente Adressen)

hoch: Z.B. wichtige "netzwerkweite" Dienste

- Bsp: NFS: Benutzung von Dateien unabhängig vom Ort
- Bsp. XWindows

Transparenz lässt sich nicht immer (einfach) erreichen

- Beispiel: Fehlertransparenz, Leistungstransparenz
- Sollte daher nicht in jedem Fall perfekt angestrebt werden (vgl. Jini)

Verteilte Systeme als “Verbunde”

- *Systemverbund*
  - gemeinsame Nutzung von Betriebsmitteln, Geräten....
  - einfache inkrementelle Erweiterbarkeit
- *Funktionsverbund*
  - Kooperation bzgl. Nutzung jeweils spezifischer Eigenschaften
- *Lastverbund*
  - Zusammenfassung der Kapazitäten
- *Datenverbund*
  - allgemeine Bereitstellung von Daten
- *Überlebensverbund*
  - i.a. nur Ausfall von Teilfunktionalität
  - Redundanz durch Replikation

Weitere Gründe für verteilte Systeme

- *Wirtschaftlichkeit*: Vernetzte Standardrechner haben i.a. besseres Preis-Leistungsverhältnis als Grossrechner
  --> Anwendung falls möglich “verteilen” (“downsizing”, outsourcing)
- *Geschwindigkeit*: Falls Anwendung “gut” parallelisierbar, ist eine sonst unerreichbare Leistung möglich
  - ggf. (dynamische) Lastverteilung beachten
- *Globalisierung* von Produktentwicklung

- Es gibt *inhärent geographisch verteilte* Systeme
  --> z.B. Zweigstellennetz einer Bank; Steuerung einer Fabrik
  --> verteilte Informationsdienste (vgl. WWW)
- *Electronic commerce*
  --> kooperative Datenverarbeitung räumlich getrennter Institutionen
  --> z.B. Reisebüros, Kreditkarten,...
- Mensch-Mensch-*Kommunikation*
  - Gruppenarbeit (“CSCW”)
  - E-mail, Sprachdienste...

Historische Entwicklung ("Systeme")

Rechner-zu-Rechner-Kommunikation

- Zugriff auf entfernte Daten (DFÜ)
- Dezentrale Informationsverarbeitung zunächst ökonomisch nicht sinnvoll (zu teuer, Fachpersonal nötig)
--> Master-Slave-Beziehung (RJE, Terminals...)

ARPA-Netz (Prototyp des Internet)

- "symmetrische" Kommunikationsbeziehung ("peer to peer")
- Internet-Protokollfamilie (TCP/IP...)
- Motivation für internationale Normung von Protokollen (OSI)
- remote login, E-mail

Workstation-Netze (LAN)

- Bahnbrechende, frühe Ideen bei XEROX-PARC (XEROX-Star als erste Workstation, Desktop-Benutzerinterface Ethernet, RPC, verteilte Dateisysteme...)
- Heute Standard bei PC-Anwendungen in Form von Produkten:
  - Kommunikation über LAN (Resource-Sharing)
  - Software für "Gruppenarbeit" (email, gem. Dateisystem...)

Forschungsprojekte

- z.B. X-Server, Kerberos,...

Kommerzielle Projekte

- z.B. Reservierungssysteme, Banken, Kreditkarten
- kooperatives Arbeiten: Workflow, "CSCW", joint authoring

WWW (und Internet) als Plattform

- für electronic commerce
- XML, peer to peer,...

"Historie" der Konzepte

- Concurrency, Synchronisation...
  - bereits klassisches Thema bei Datenbanken und Betriebssystemen
- Programmiersprachen
  - kommunizierende Objekte, CSP...
- Physische Parallelität
  - Array-, Pipeline-, Multiprozessoren
- Parallele und verteilte Algorithmen
- Semantik
  - math. Modelle, CCS, Petri-Netze...
- Abstraktionsprinzipien
  - Schichten, Dienstprimitive,...
- Verständnis grundlegender Phänomene der Verteiltheit
  - Konsistenz, Zeit, Zustand...

Entwicklung "guter" Konzepte, Modelle, Abstraktionen etc. zum Verständnis der Phänomene dauert oft lange

- notwendige Ordnung und Sichtung des verfügbaren Gedankenguts

Diese sind jedoch für die Lösung praktischer Probleme hilfreich, gelegentlich sogar notwendig!

Charakteristika und "praktische" Probleme verteilter Systeme

- Räumliche Separation, autonome Komponenten

--> Zwang zur Kommunikation per Nachrichtenaustausch

--> Neue Probleme:

- partielles Fehlverhalten (statt totaler "Absturz")
- fehlender globaler Zustand / globale Zeit
- Inkonsistenzen, z.B. zwischen Datei und Verzeichnis
- Konkurrenter Zugriff, Replikate, Cache,...

Eingesetzt zur Realisierung von Leistungs- und Ausfalltoleranz

- Heterogenität
  - Ist in gewachsenen Informationsumgebungen eine Tatsache
  - Findet sich in Hard- und Software

- Dynamik, Offenheit
  - "Interoperabilität" zu gewährleisten ist nicht einfach

- Komplexität
  - Verteilte Systeme schwierig zu entwickeln, betreiben, beherrschen
  - Abstraktion als Mittel zur Beherrschung von Komplexität wichtig:

    a) Schichten (Kapselung, virtuelle Maschinen...)
    b) Modularisierung (Services, Mikrokerne...)
    c) "Transparenz"-Prinzip

- Sicherheit
  - Vertraulichkeit, Authenzitität, Integrität, Verfügbarkeit...
  - notwendiger als in klassischen Systemen
  - aber schwieriger zu gewährleisten (mehr Schwachstellen)

Aspekte verteilter Systeme

im Vergleich zu *sequentiellen* Systemen:

- Grösse und Komplexität
- Heterogenität → jede(r) ist anders
- Nebenläufigkeit → vieles gleichzeitig
- Nichtdeterminismus → morgen anders als heute...
- Zustandsverteilung → niemand weiss alles

- Programmierung *komplexer*
- Test und Verifikation *aufwendiger*
- Verständnis der Phänomene *schwieriger*

==> gute Werkzeuge ("Tools") und Methoden

- z.B. Middleware als Software-Infrastruktur

==> adäquate Modelle, Algorithmen, Konzepte

- zur Beherrschung der neuen Phänomene

Ziel: Verständnis der grundlegenden Phänomene, Kenntnis der geeigneten Konzepte und Verfahren

Einige *konzeptionelle* Probleme und Phänomene verteilter Systeme

1) Schnappschussproblem

2) Phantom-Deadlocks

3) Uhrensynchronisation

4) Kausaltreue Beobachtungen

5) Geheimnisvereinbarung über unsichere Kanäle

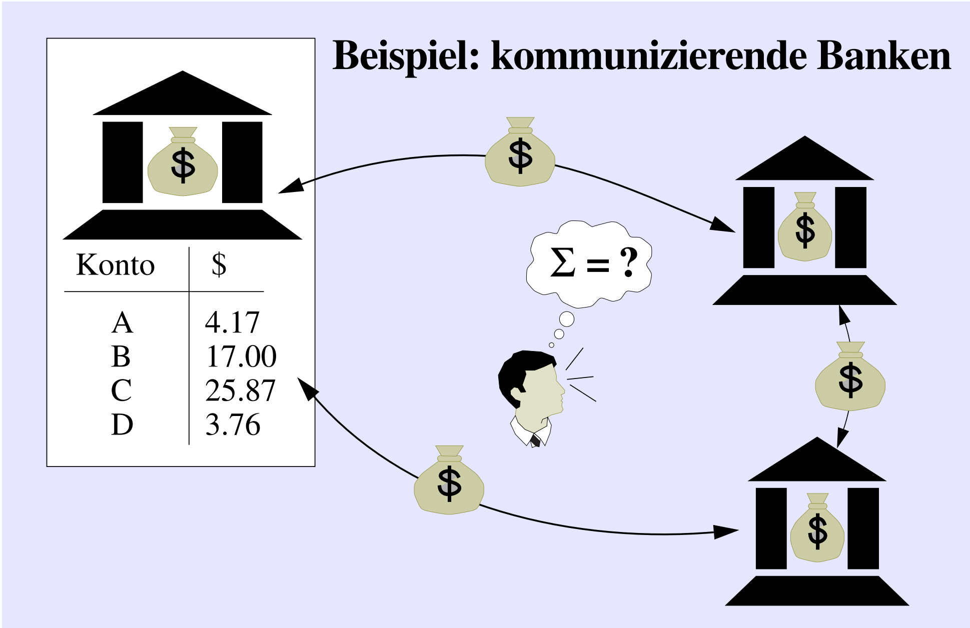
- Dies sind einige einfach zu erläuternde Probleme und Phänomene

- Es gibt noch viel mehr und viel komplexere Probleme
    - konzeptioneller Art
    - praktischer Art

- Achtung: Manches davon wird nicht hier, sondern in der Vorlesung “Verteilte Algorithmen” behandelt!

Ein erstes Beispiel: Wieviel Geld ist in Umlauf?

- konstante Geldmenge, oder
- monotone Inflation (--> Untergrenze)

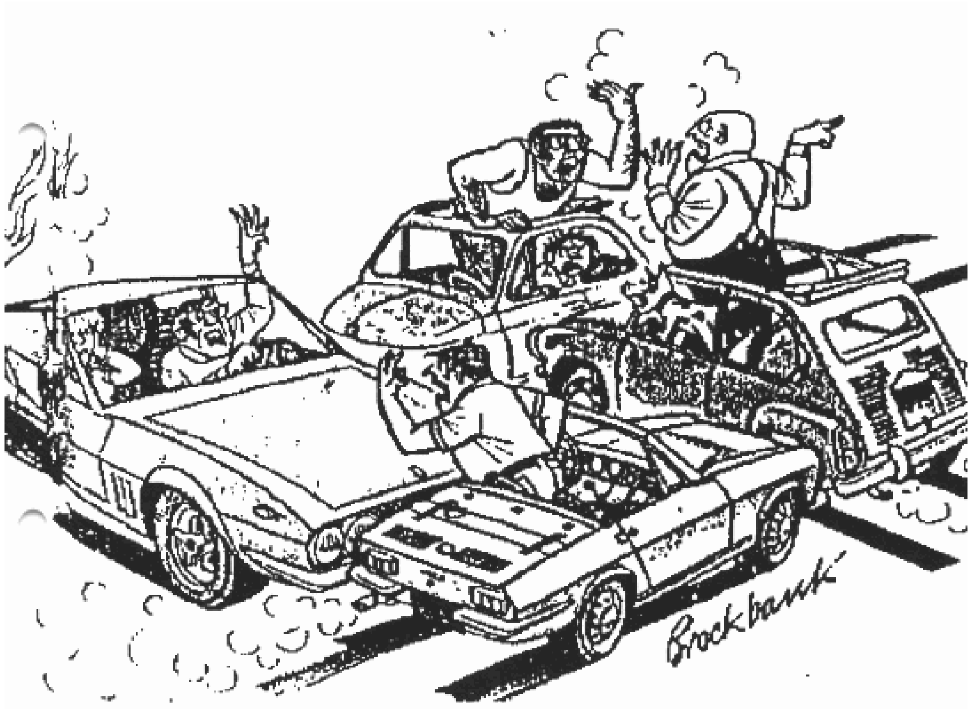


- Modellierung:
    - verteilte Geldkonten
    - ständige Transfers zwischen den Konten

- Erschwerte Bedingungen:
    - niemand hat eine globale Sicht
    - es gibt keine gemeinsame Zeit (“Stichtag”)

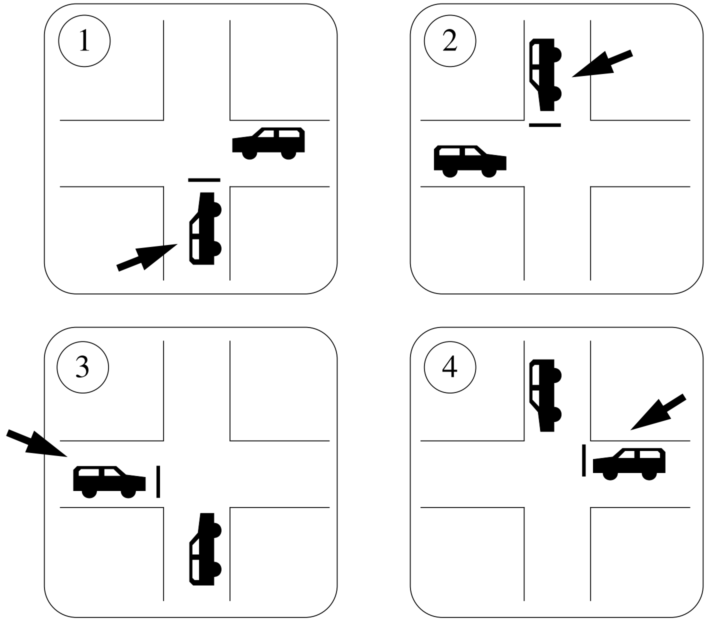
- Anwendung: z.B. verteilte DB-Sicherungspunkte

Ein zweites Beispiel: Das Deadlock-Problem



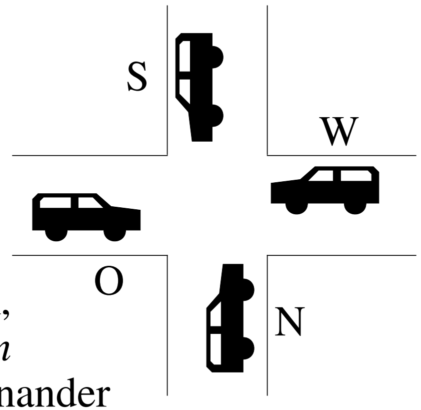
„Neapolitanischer Hakenkreuzstau“
(Also sprach Bellavista, Luciano De Crescenzo)

Phantom-Deadlocks



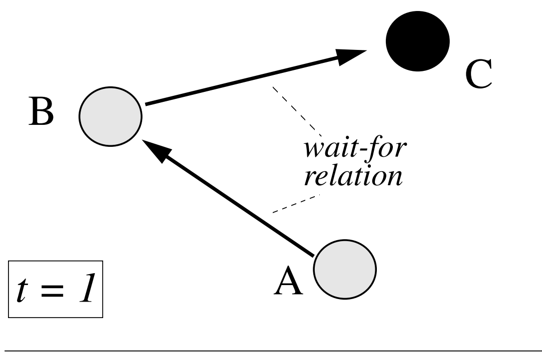
Vier Einzelbeobachtungen der Autos N, S, O, W

1) N wartet auf W
2) S wartet auf O
3) O wartet auf N
4) W wartet auf S



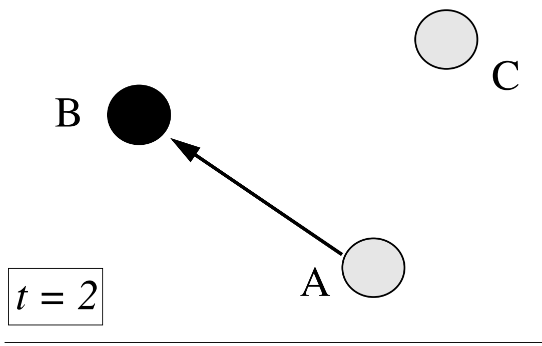
zu notwendigerweise verschiedenen Zeitpunkten liefert den *falschen* Eindruck, als würden zu einem *einzigen* Zeitpunkt alle zyklisch aufeinander warten (--> Verklemmung)

Phantom-Deadlocks

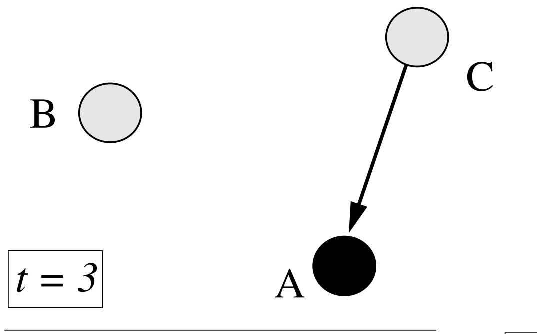


(C benutzt ein exklusives Betriebsmittel)

beobachte B:
==> B wartet auf C

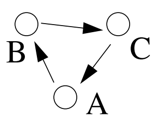


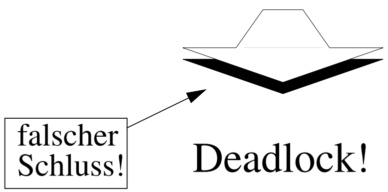
beobachte A:
==> A wartet auf B



beobachte C:
==> C wartet auf A

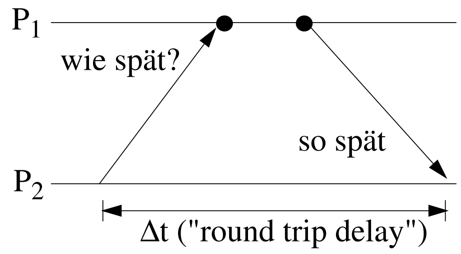
Keine exakte globale Zeit!



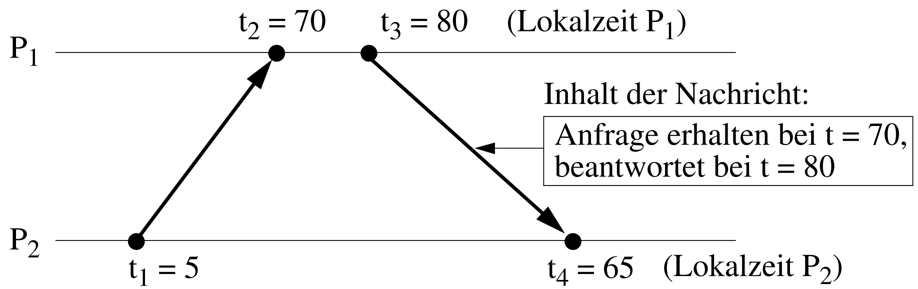


Deadlock!

Ein drittes Problem: Uhrensynchronisation



- Lastabhängige Laufzeiten von Nachrichten
- Unsymmetrische Laufzeiten
- Wie erfährt man die Laufzeit?

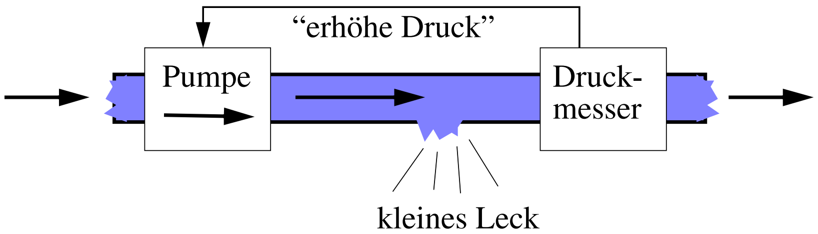


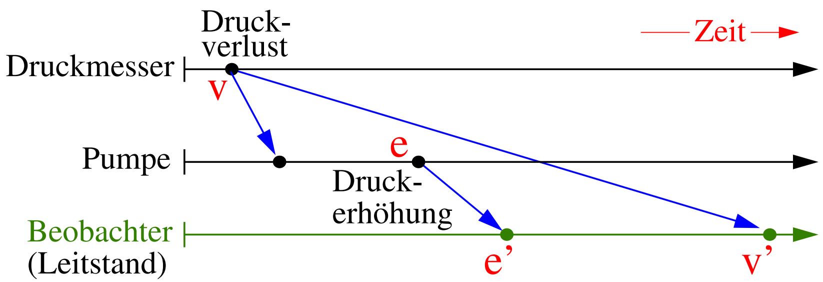
- Uhren gehen nicht unbedingt gleich schnell!
  (wenigstens “Beschleunigung ≈ 0”, d.h. konstanter Drift gerechtfertigt?)

- Wie kann man den Offset der Uhren ermitteln oder zumindest approximieren?

Ein viertes Problem: Kausal (in)konsistente Beobachtungen

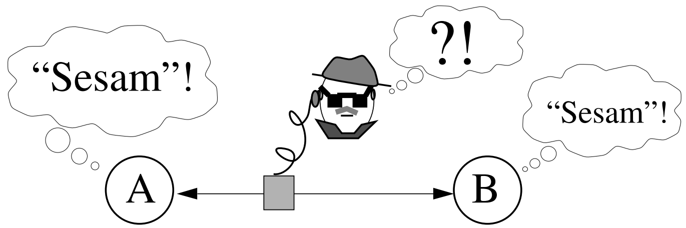
- Gewünscht: Eine Ursache stets vor ihrer (u.U. indirekter) Wirkung beobachten





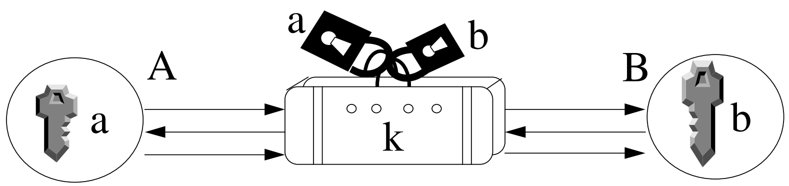
Falsche Schlussfolgerung des Beobachters:
Eine unbegründete Pumpenaktivität erhöhte den Druck bis zum Bersten der Pipeline; daraufhin trat das Öl aus dem Leck aus, was durch den Druckverlust angezeigt wird!

Und noch ein Problem: Verteilte Geheimnisvereinbarung



- Problem: A und B wollen sich über einen unsicheren Kanal auf ein gemeinsames geheimes Passwort einigen.

- Idee: Vorhängeschlösser um eine sichere Truhe:



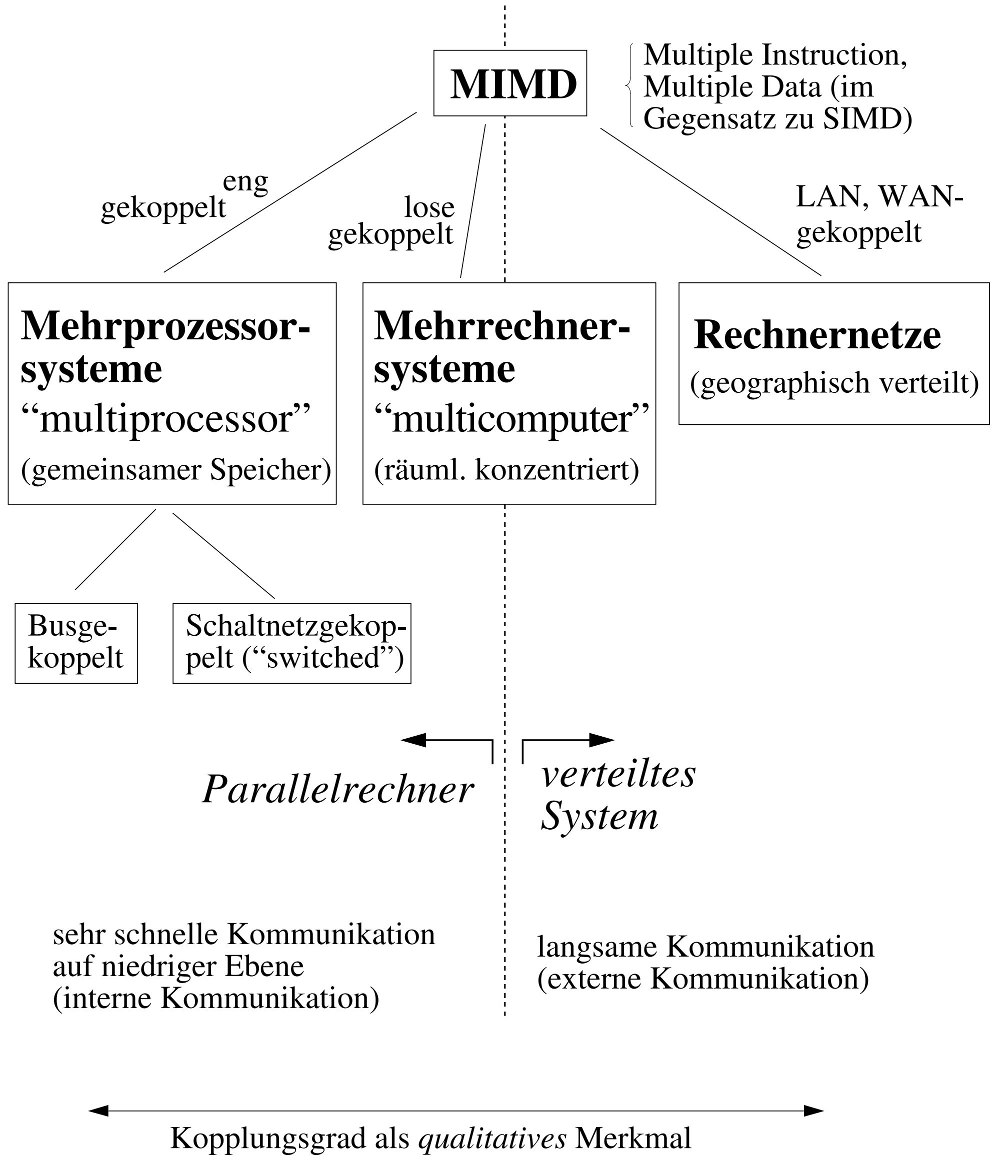
1. A denkt sich Passwort k aus und tut es in die Truhe.
2. A verschliesst die Truhe mit einem Schloss a.
3. A sendet die so verschlossene Truhe an B.
4. B umschliesst das ganze mit seinem Schloss b.
5. B sendet alles doppelt verschlossen an A zurück.
6. A entfernt Schloss a.
7. A sendet die mit b verschlossene Truhe wieder an B.
8. B entfernt sein Schloss b.

- Problem: Lässt sich das so softwaretechnisch realisieren?

Wie wäre es damit?: k sei eine Zahl. "Verschliessen" und "aufschliessen" eines Schlosses entspricht dem Hinzuaddieren oder Subtrahieren einer beliebig ausgedachten (geheimgehaltenen) Zahl a bzw. b.

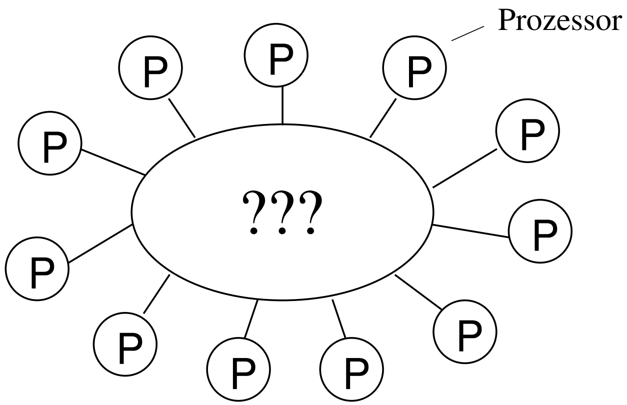
Multiprozessoren und Multicomputer

Abgrenzung Parallelrechner



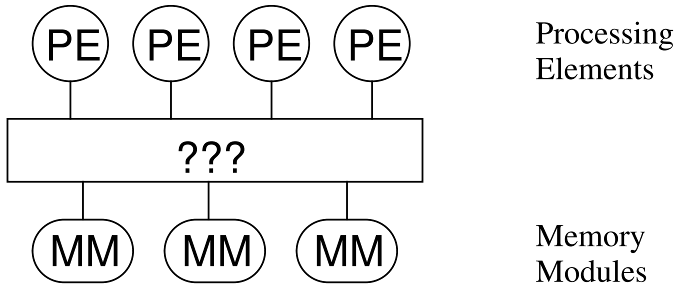
Prozessorverbund

- Autonome Prozessoren + „Kommunikationsnetz“
- Je nach Kopplungsgrad und Grad der Autonomie ergibt sich daraus ein
  - Mehrprozessorsystem
  - Mehrrechnersystem
  - Rechnernetz



Speicherkopplung

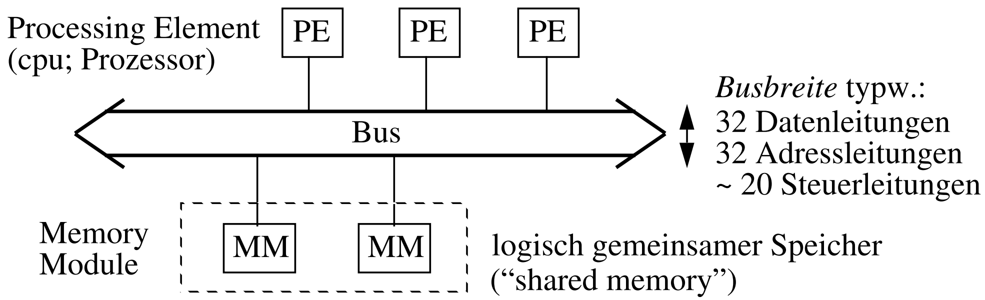
- Shared Memory
  - Kommunikation über gemeinsamen Speicher



- n Processing Elements teilen sich k Memory Modules
- Kopplung zwischen PE und MM, z.B.
  - Bus
  - Schaltnetz
  - Permutationsnetz
- UMA-Architektur (Uniform Memory Access) oder NUMA (Non-Uniform Memory Access)

wenn es “nahe” und “ferne” Speicher gibt: z.B. schneller Zugriff auf den “eigenen” Speicher, langsamer auf fremden

Busgekoppelte Multiprozessoren

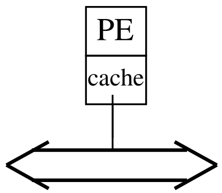


Problem:

Bus i.a. bereits bei wenigen (3 - 5) PEs überlastet

Lösung:

Lokale Caches
zwischen PE und Bus:



Cache gross genug wählen, um Hitraten > 90% zu erzielen (abhängig von der Hauptspeichergrösse)!

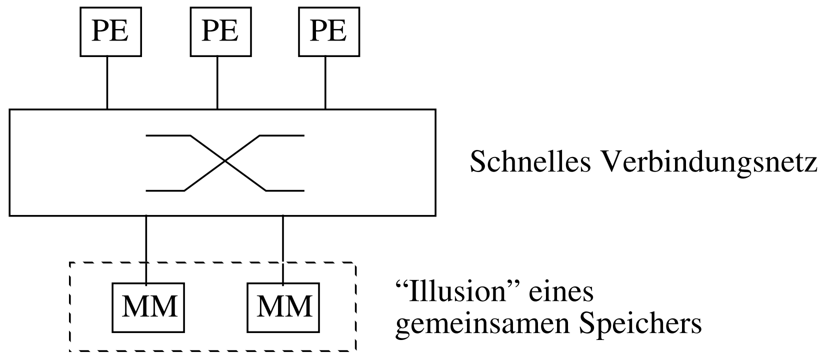
Probleme:

1) Kohärenzproblem der caches

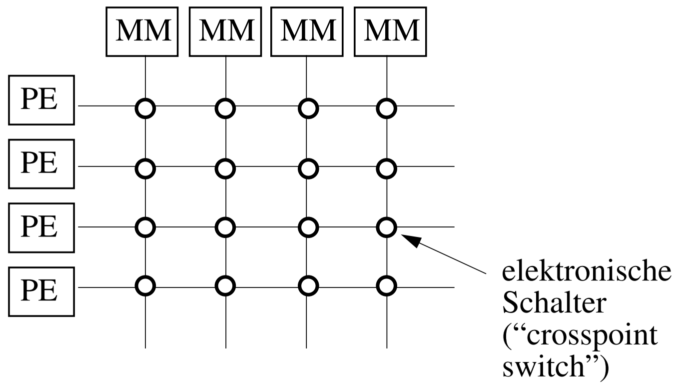
2) Damit Problem nur verschoben
(ca. 10 Mal mehr Prozessoren möglich)

Generell: Busgekoppelte Systeme schlecht skalierbar!
(Übertragungsbandbreite bleibt "konstant" bei Erweiterung um Knoten)

Schaltnetzgekoppelte Multiprozessoren



Z.B. Crossbar-switch (Kreuzschienenverteiler):

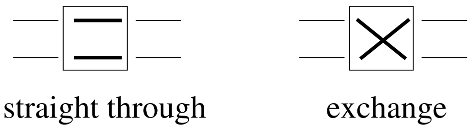


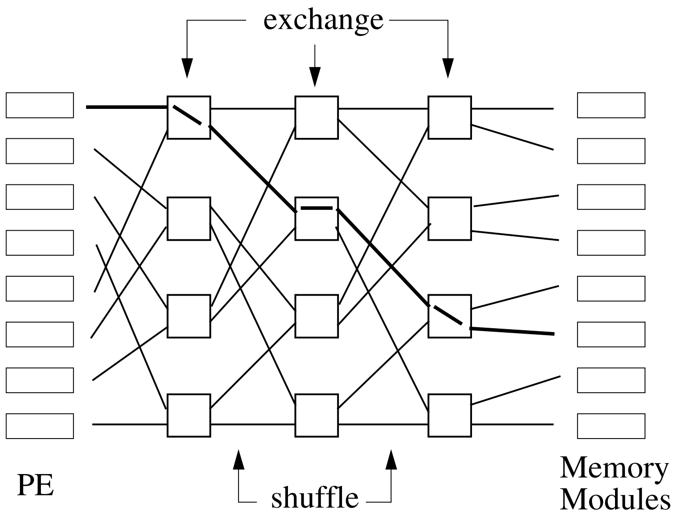
- Mehrere PEs können gleichzeitig auf verschiedene Speichermodule zugreifen
- Schlecht skalierbar (quadratisch viele Schalter)
(Vermeidung von hot spots durch interleaving, Randomisierung...)

Permutationsnetze

Mehrere Stufen von Schaltelementen ermöglichen die Verbindung jedes Einganges zu jedem Ausgang.

Schaltelement (“interchange box”) kann zwei Zustände annehmen (durch ein Bit ansteuerbar):





Beispiel:
Shuffle-Exchange-
Netz

(Omega-
Netz)

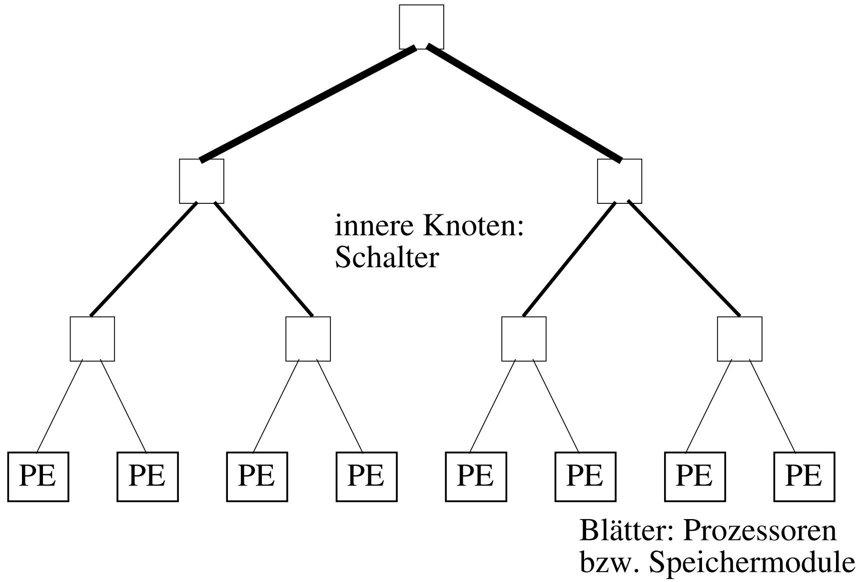
Hier: log n (identische!) Stufen mit je n/2 Schaltern.

Es gibt weitere ähnliche dynamisch schaltbare Netze. Designkriterien:

z.B. Butterfly-Netze

- wenig Stufen (“delay”)
- Parallele Zugriffe; Vermeidung von Blockaden
- ggf. Zugriffe bündeln (“combining”)

Fat-Tree-Netze



Verbindungsleitungen höherer Bandbreite bzw. mehrere parallele Leitungen auf Niveaus, die näher an der Wurzel liegen.

Multiprozessoren — Fazit

- Gemeinsamer Speicher, über den die Prozessoren Information austauschen (d.h. kommunizieren) können
    - Prozessoren müssen mit dem Speicher (bzw. den einzelnen Speichermodulen) gekoppelt werden

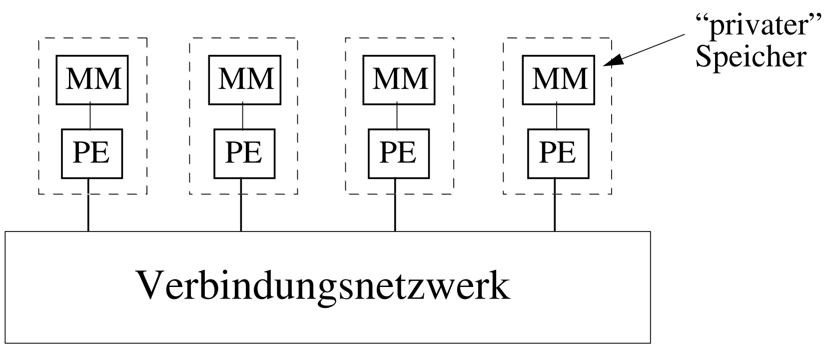
- Speicherkopplung begrenzt Skalierbarkeit und räumliche Ausdehnung
    - Untergliederung des Speichers in mehrere Module (Parallelität)
    - leistungsfähiges Kommunikationsnetz

- Lokale PE-Caches sinnvoll
    - Problem der Cache-Kohärenz

- Bewertungskriterien für Verbindungsnetze
    - Realisierungsaufwand (Fläche, Kosten)
    - Skalierbarkeit (mit wachsender Anzahl PEs und MMs)
    - innere Blockadefreiheit (parallele Kommunikationsvorgänge)
    - Anzahl der Stufen (Verzögerung)
    - Eingangsgrad, Ausgangsgrad der Bauelemente

Mehrrechnersysteme (“Multicomputer”)

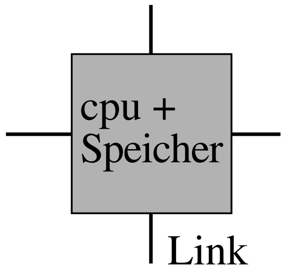
Vernetzung vollständiger Einzelrechner:



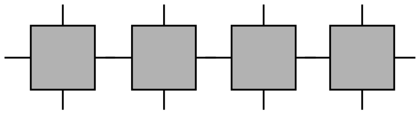
Zugriff auf andere Rechner (bzw. deren private Speicher) nur indirekt über *Nachrichten*.

- kein globaler Speicher
- NORMA-Architektur (NO Remote Memory Access)

Beispiel: Transputer als Baustein für Multicomputer

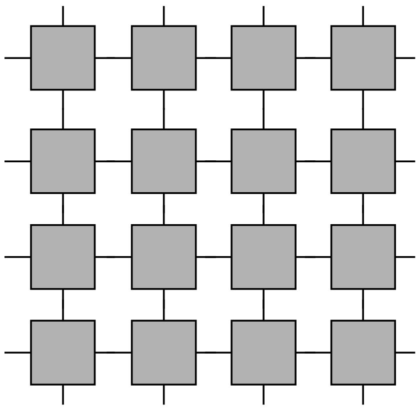


- Typische Topologien:



Pipeline

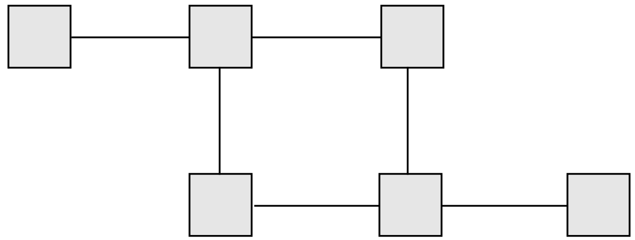
Array (oder Torus)



- bidirektional über mehrere Meter
- kleine set-up time
- synchrone Komm.
- Mehrprozesskonzept mit Scheduler "on chip"
- Eigene Programmiersprache "Occam"

- Hersteller: INMOS (GB)
- Erste Modelle: 1983;   T414 (1986), T800 (1987)
  - in den 90er-Jahren keine Nachfolgetypen mehr entwickelt

Verbindungstopologien für Mehrrechnersysteme



Zusammenhängender Graph mit

Knoten = Rechner

Kante = dedizierte Kommunikationsleitung

Ausdehnung: i.a. nur wenige Meter

Bewertungskriterien:

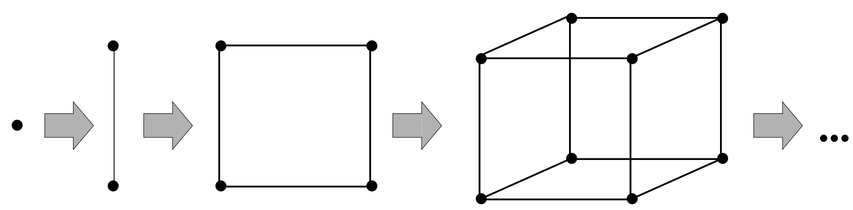
- Gesamtzahl der Verbindungen (bei n Knoten)
- maximale Entfernung zweier Knoten
- durchschnittliche Entfernung
- Anzahl der Nachbarn eines Knotens ("fan out")
- Symmetrie, Homogenität, Skalierbarkeit...
- Routingkomplexität
- Zahl der alternativ bzw. parallel verfügbaren Wege

Technologische Faktoren:

- Geschwindigkeit, Durchsatz, Verzögerung, eigene Kommunikationsprozessoren...

Hypercube

- Hypercube = "Würfel der Dimension d"

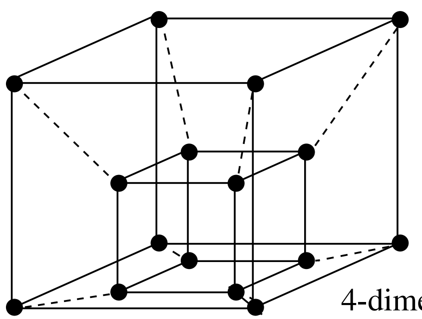


Draufsicht von der Seite liefert jeweils niedrigere Dimension

Entsprechend: Herausdrehen des Objektes aus der Blickebene zeigt, dass es sich "eigentlich" um ein Objekt der Dimension n+1 handelt!

- Rekursives Konstruktionsprinzip
  - Hypercube der Dimension 0: Einzelrechner
  - Hypercube der Dimension d+1:

    „Nimm zwei Würfel der Dimension d und verbinde korrespondierende Ecken“

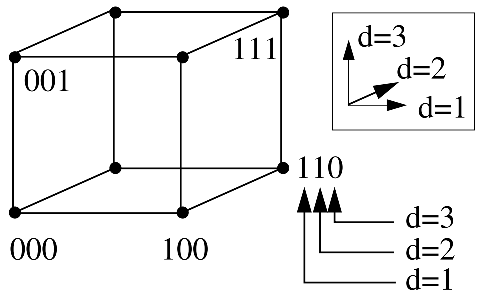


4-dimensionaler Würfel

Man vgl. auch das Buch von T. F. Banchoff: Beyond the Third Dimension (Scientific American Library, 1990)

Hypercube der Dimension d

- $n = 2^d$ Knoten
- Anzahl der Nachbarn eines Knotens = d
  (Anzahl der "ports" in der Hardware)
- Gesamtzahl der Kanten (= Verbindungen): $d\ 2^d/2 = d\ 2^{d-1}$
  (Ordnung $O(n \log n)$)
- Einfaches Routing:
  - Knoten systematisch (entspr. rekursivem Aufbau) numerieren
  - Zieladresse bitweise xor mit Absenderadresse
  - Wo sich eine "1" findet, in diese Dimension muss gewechselt werden



- Maximale Weglänge: d
- Durchschnittliche Weglänge = d/2
  (Induktionsbeweis als Übung!)

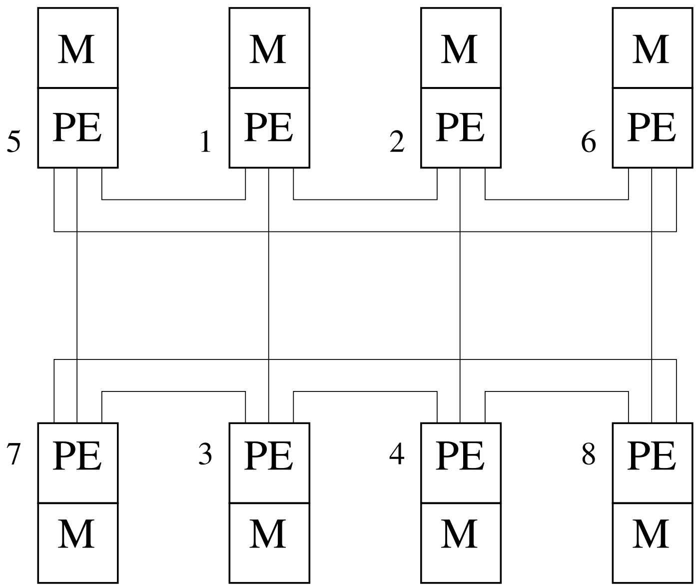
-Vorteile Hypercube:
  - kurze Weglängen (max. log n)
  - einfaches Routing
  - viele Wegalternativen (Fehlertoleranz, Parallelität!)

Denkübung: mittlere Weglänge?

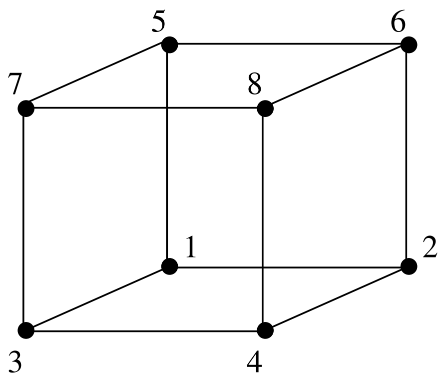
wieviele verschiedene Wege der Länge k gibt es insgesamt?

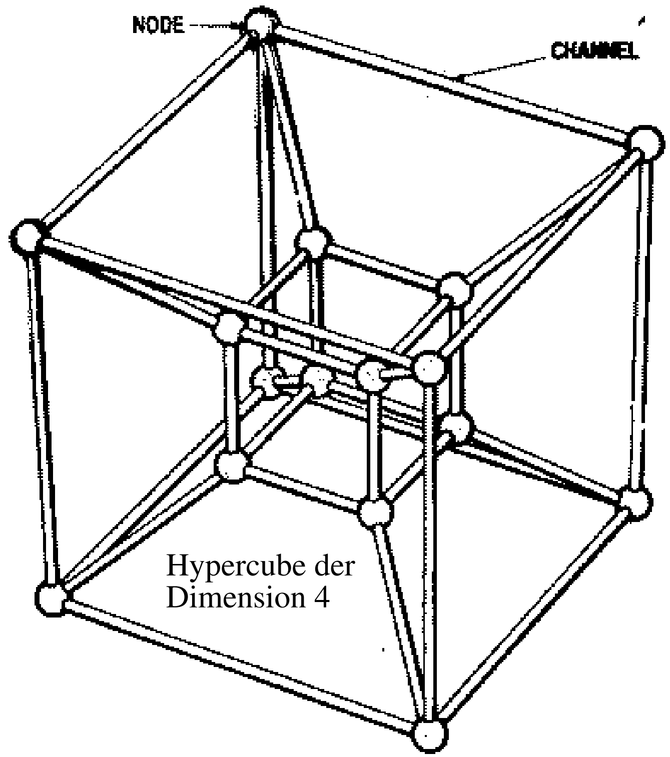
-Nachteile:
  - Anzahl der Nachbarknoten eines Knotens wächst mit der Dimension d
  - insgesamt relativ viele Verbindungen: O(n log n) (eigentlich genügen n-1)

Layout eines Hypercube

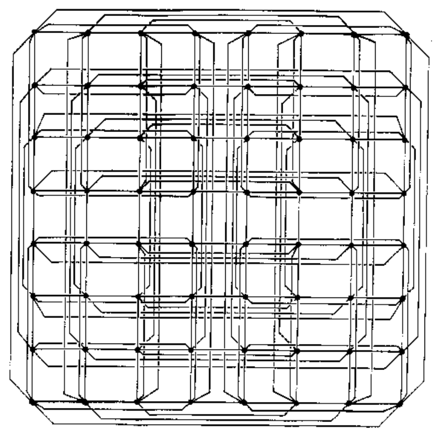


Obiger Topologie sieht man zunächst nicht an, dass es sich dabei um einen 3-dimensionalen Würfel handelt!





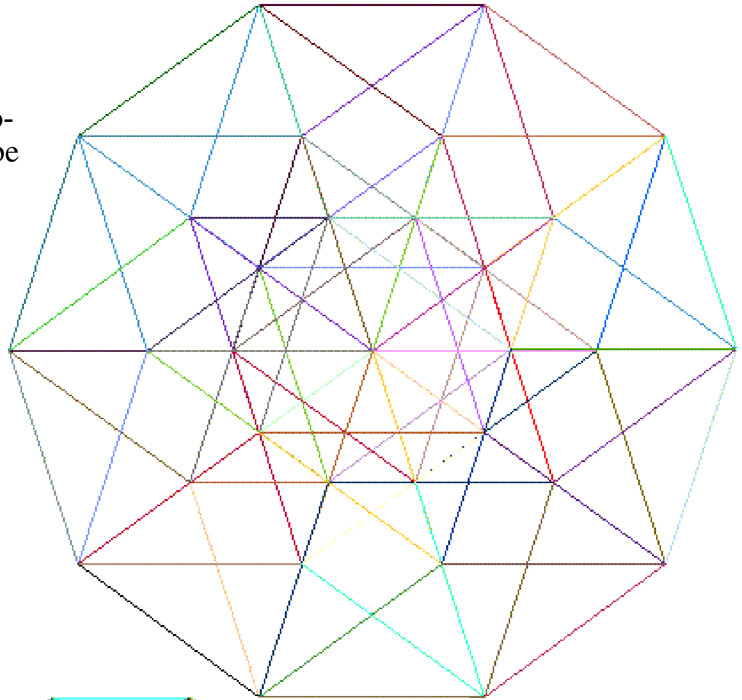
Hypercube der Dimension 4

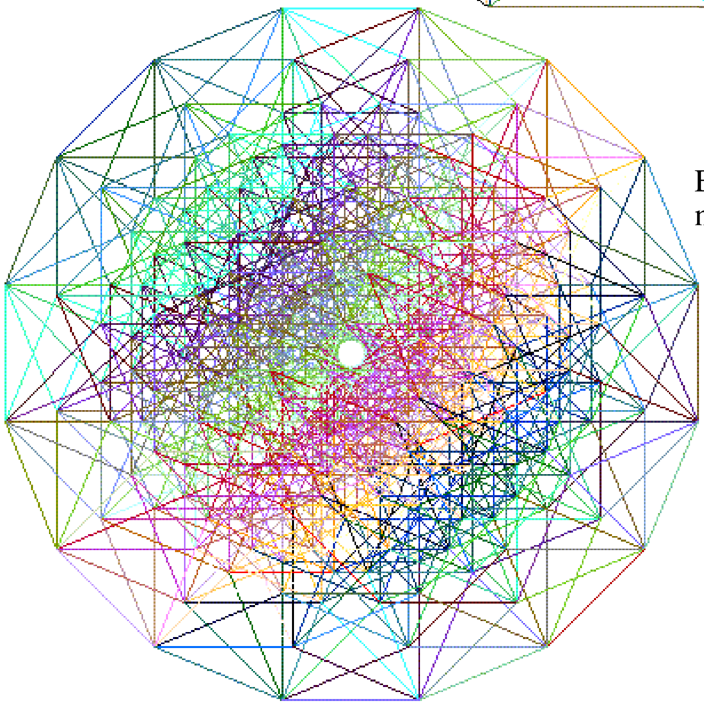


Hypercube der Dimension 6 in der Ebene

Ein 5-dimensionaler Hypercube

www.cs.reading.ac.uk/archive/hypercubes/





Ein 8-dimensionaler Hypercube

www.comlab.ox.ac.uk/oucl/users/jonathan.bowen/publications/ndcubes.html

Broadcast in Hypercubes

- Broadcast: Eine Nachricht an alle anderen Knoten senden.
- Initiator habe die Nummer 00...00 (binär).
- Initiator sendet an alle seine Nachbarn:

am besten gleichzeitig, wenn dies technisch geht!

$\underbrace{\text{0...01, 0...010, 0...100, ... , 10...0}}_{\text{in “kanonischer” Numerierung}}$

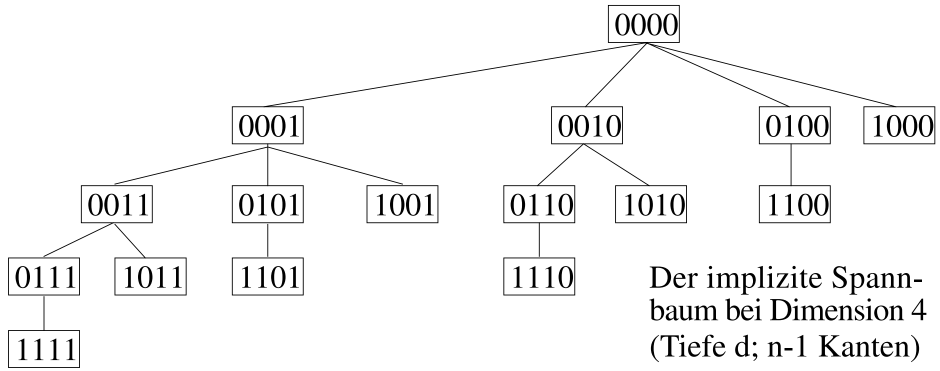
linkeste 1 | beliebiges Restmuster

- Ein Knoten mit der Nummer 0...01x...y...z leitet die Information an alle seine “höheren” Nachbarn weiter:

0...00**1**1x...y...z
0...0**1**01x...y...z
0...**1**001x...y...z
...
10...001x...y...z

Von welchem (eindeutigen) Knoten A wird Knoten B informiert?

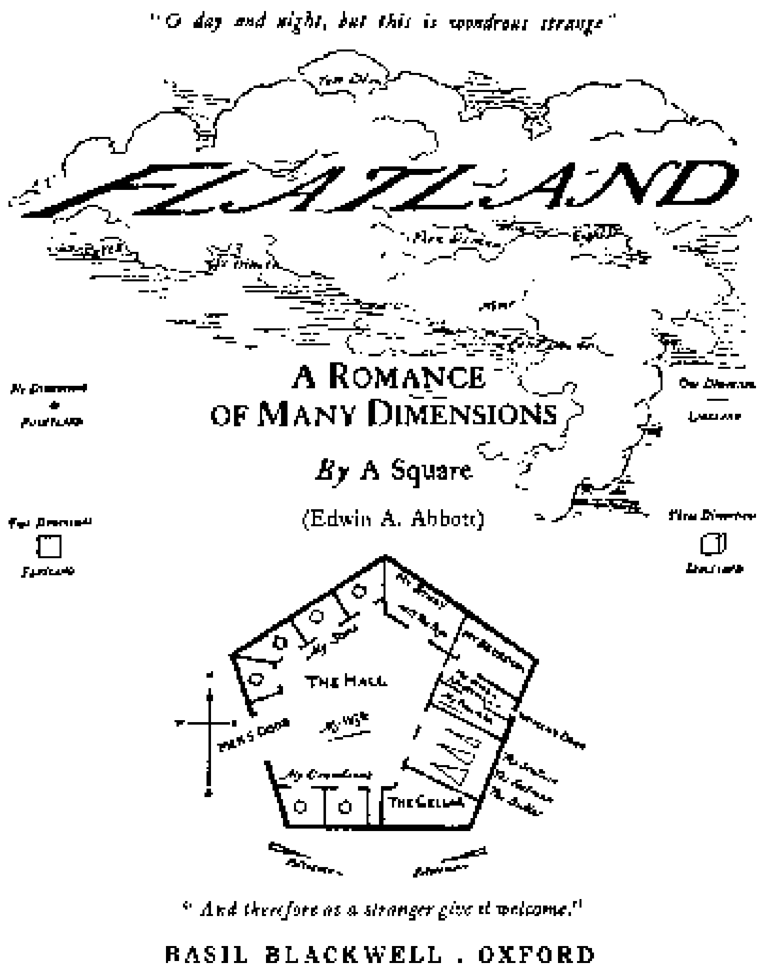
Setze *vorderste 1* von B auf 0 --> = Nummer von A



Der implizite Spannbaum bei Dimension 4 (Tiefe d; n-1 Kanten)

- Der Algorithmus wird in der Praxis (z.B. Mehrprozessorsysteme) verwendet
- Wie effizient ist der Algorithmus? (Geht es besser?)
- Denkübung: Formuliere Algorithmus für einen beliebigen Initiator (schliesslich sind Hypercubes symmetrisch...)
- Andere (bessere?) Algorithmen in der Vorlesung “Verteilte Algorithmen”

Flatland (1884)



- Diskussion über Hypercubes und höhere geometrische Dimensionen zwischen "A. Square" und "Sphere"

- Gleichzeitig soziale Satire

§ 3 .--- *Concerning the Inhabitants of Flatland*

...

Our Women are Straight Lines.

Our Soldiers and Lowest Class of Workmen are Triangles with two equal sides, each about eleven inches long ...

Our Middle Class consists of Equilateral or Equal-Sided Triangles.

Our Professional Men and Gentlemen are Squares (to which class I myself belong) and Five-Sided Figures or Pentagons.

Next above these come the Nobility, of whom there are several degrees, beginning at Six-Sided Figures, or Hexagons, and from thence rising in the number of their sides till they receive the honourable title of Polygonal, or many-Sided. Finally when the number of the sides becomes so numerous, and the sides themselves so small, that the figure cannot be distinguished from a circle, he is included in the Circular or Priestly order; and this is the highest class of all.

...

§ 4 .--- *Concerning the Women*

If our highly pointed Triangles of the Soldier class are formidable, it may be readily inferred that far more formidable are our Women.

...

But here, perhaps, some of my younger Readers may ask HOW a woman in Flatland can make herself invisible. This ought, I think, to be apparent without any explanation. However, a few words will make it clear to the most unreflecting.

Place a needle on the table. Then, with your eye on the level of the table, look at it side-ways, and you see the whole length of it; but look at it end-ways, and you see nothing but a point, it has become practically invisible. Just so is it with one of our Women.

...

The dangers to which we are exposed from our Women must now be manifest to the meanest capacity of Spaceland. If even the angle of a respectable Triangle in the middle class is not without its dangers ... --what can it be to run against a woman, except absolute and immediate destruction? And when a Woman is invisible, or visible only as a dim sub-lustrous point, how difficult must it be, even for the most cautious, always to avoid collision!

...

In the Southern and less temperate climates, where the force of gravitation is greater, and human beings more liable to casual and involuntary motions, the Laws concerning Women are naturally much more stringent. But a general view of the Code may be obtained from the following summary:--

1. Every house shall have one entrance on the Eastern side, for the use of Females only; by which all females shall enter "in a becoming and respectful manner" and not by the Men's or Western door.

2. No Female shall walk in any public place without continually keeping up her Peace-cry, under penalty of death.

3. ...

In some of the States there is an additional Law forbidding Females, under penalty of death, from walking or standing in any public place without moving their backs constantly from right to left so as to indicate their presence to those behind them...

§ 16.---*How the Stranger vainly endeavoured to reveal to me in words the mysteries of Spaceland*

...

Sphere. ... We began with a single Point, which of course -- being itself a Point -- has only ONE terminal Point. One Point produces a Line with TWO terminal Points.

One Line produces a Square with FOUR terminal Points.

Now you can give yourself the answer to your own question: 1, 2, 4, are evidently in Geometrical Progression. What is the next number?

I. Eight.

Sphere. Exactly. The one Square produces a SOMETHING-WHICH-YOU-DO-NOT-AS-YET-KNOW-A-NAME-FOR-BUT-WHICH-WE-CALL-A-CUBE with EIGHT terminal Points. Now are you convinced?

...

Sphere. How can you ask? And you a mathematician! The side of anything is always, if I may so say, one Dimension behind the thing. Consequently, as there is no Dimension behind a Point, a Point has 0 sides; a Line, if I may so say, has 2 sides (for the points of a Line may be called by courtesy, its sides); a Square has 4 sides; 0, 2, 4; what Progression do you call that?

I. Arithmetical.

Sphere. And what is the next number?

I. Six.

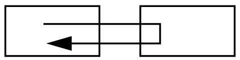
Sphere. Exactly. Then you see you have answered your own question. The Cube which you will generate will be bounded by six sides, that is to say, six of your insides. You see it all now, eh?

"Monster," I shrieked, "be thou juggler, enchanter, dream, or devil, no more will I endure thy mockeries. Either thou or I must perish." And saying these words I precipitated myself upon him.

Online-Text erhältlich bei: http://ebbs.english.vt.edu/20th/txts/abbott/guten.boiler.html oder: http://wiretap.spies.com/ftp.items/Library/Classic/flatland.txt

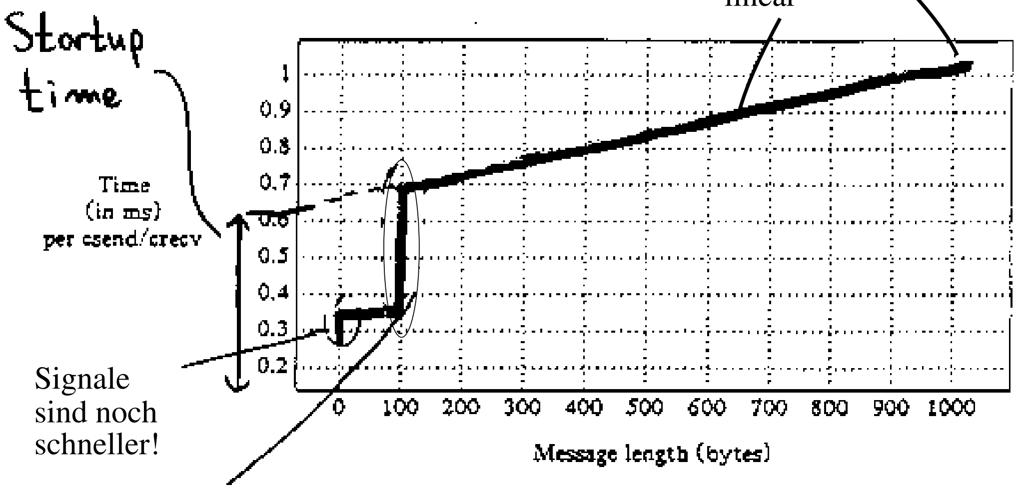
Das *Buch* ist erhältlich in mehreren Ausgaben; z.B.: Abbott, Edwin A.: Flatland. Penguin, 1987, ISBN: 0140076158, DM 11,90

Die Intel-Hypercube-Parallelrechner

- Forschungsprototyp “Cosmic Cube”: Caltech (vor 1985)
- iPSC/1 ca. ab 1985; iPSC/2 ca. ab 1988, dann iPSC/860
- Typische Grössen: 16, 32, 64, 128 Knoten
  - Teilwürfel sind unabhängig nutzbar
- Spätere Maschinen nutzen Gittertopologie statt Hypercube!
- Hier: Messung der Kommunikationsleistung
  - Technik: Round-trip-Zeit 

1 MB/s ?

linear



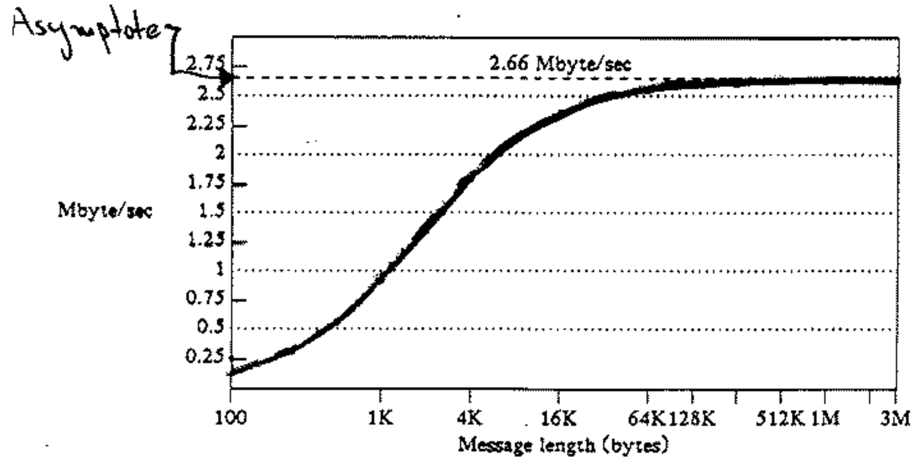
Signale sind noch schneller!

Bei Nachrichten > 100 Byte: zunächst überprüfen, ob der Empfänger genügend Pufferkapazität hat!

Man lese dazu folgenden Artikel: L. Bomans and D. Roose, *Benchmarking the iPSC/2 Hypercube Multiprocessor*, Concurrency - Practice and Experience, Vol.1 No 1, pp. 3-18, 1989

Asymptotische Übertragungsrate

- Konsequenz der relativ grossen Startup-Zeit beim iPSC: 2.8 MB/s “peak performance” ist ein asymptotischer Wert; er gilt nicht für “normale” Nachrichten!



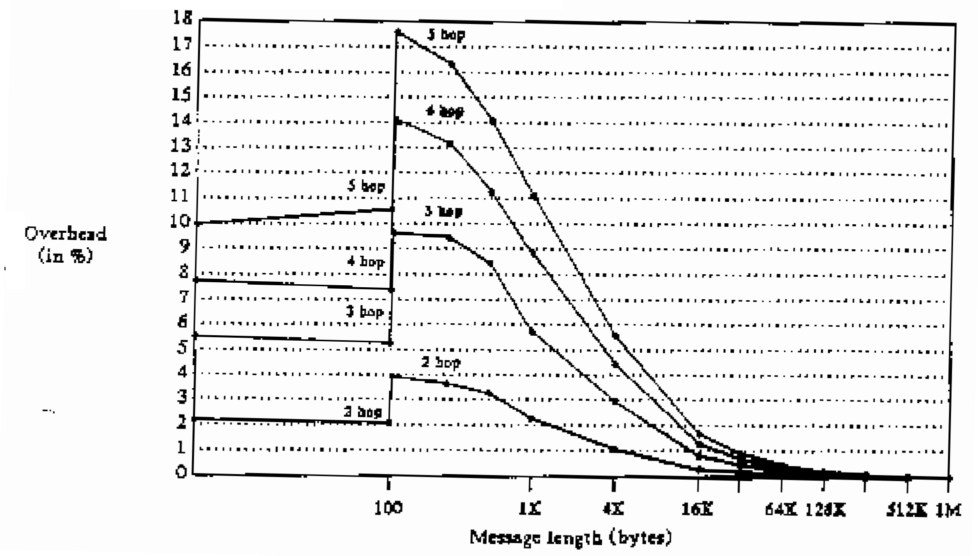
- Konstanter Overhead amortisiert sich erst bei sehr grossen Nachrichten

Multi-hop-Kommunikation

Kommunikation zwischen nicht benachbarten Knoten: Wie schnell ist die Kommunikation dabei beim iPSC?

--> Zusatzaufwand fast vernachlässigbar!

iPSC/2-Hypercube der Dimension 6 --> maximal 5 “hops”

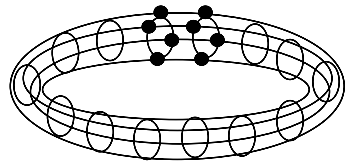


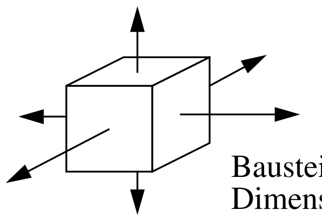
Weitere Frage:

Gegenseitige Behinderung verschiedener Nachrichten wegen gemeinsam benutzter Verbindungen? (“Channel contention”)

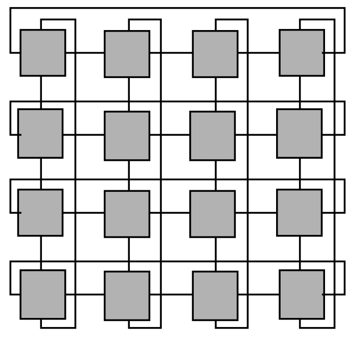
Eine andere Verbindungstopologie: der d-dimensionale Torus

= d-dimensionales “wrap-around Gitter”





Baustein für 3 Dimensionen



2 Dimensionen

- Rekursives Konstruktionsprinzip: „Nimm w_{d-1} gleiche Tori der Dimension d -1 und verbinde korrespondierende Elemente zu Ring“

- Bei Ausdehnung w_i in Dimension i:
  $n = w_1 \times w_1 \times .. \times w_d$ Knoten;
  mittlere Entfernung zw. 2 Knoten: $\Delta \approx \frac{1}{4} \sum w_i$

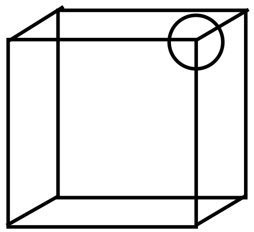
- *Ring* als Sonderfall d = 1 !

- *Hypercube* der Dimension d ist d-dimensionaler Torus mit $w_i = 2$ für alle Dimensionen!

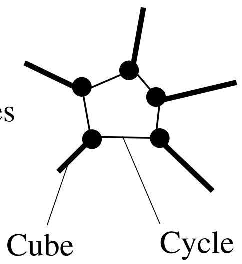
--> $\Delta = \frac{1}{4} \sum_d 2 = \frac{1}{4}(2\,d) = \frac{d}{2} = \frac{1}{2}\log_2 n$

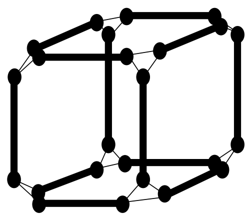
Cube Connected Cycles (CCC)

d-dimensionaler Hypercube mit aufgeschnittenen Ecken, die durch Gruppen von d ringförmig verbundenen Knoten ersetzt sind.



"Ecke" eines CCC (d=5)





Beachte: Jeder Knoten hat immer *drei* Anschlüsse!

Bei Dimension d: $n = d\ 2^d$

Maximale / mittlerer Weglänge?

Denkübung! (wieso < 2d ?)

Anzahl der Verbindungen = 3 n / 2
(statt O(n log n) wie beim Hypercube)

Es gibt viele weitere Verbindungstopologien...
(z.B. de Brujn Graphen)

Zufallstopologien (mit max. Grad = 4)

Verfahren A:

1) Starte mit einem Graphen ohne Kanten.
2) Wähle zwei zufällige Knoten, die noch weniger als 4 Kanten haben, und verbinde diese.
3) Wiederhole Schritt 2 solange wie möglich.
4) Falls ein unzusammenhängender Graph entsteht, beginne von vorne.

Verfahren B ("greedy graphs"):

Motivation: Kurze Zyklen vermeiden

2) Wähle zwei bel. Knoten *mit maximaler Entfernung* (einschliesslich ∞), die noch weniger als 4 Kanten haben, und verbinde diese.

Wie gut sind diese Zufallsgraphen bzgl. mittlerer Knotenentfernung und Routingbelastung?

Verzögerungszeiten, Routingoverhead

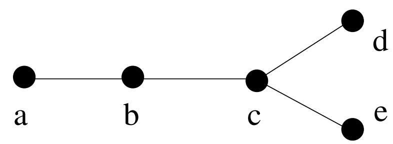
Bottleneck

Bem.: Explizites Routing nötig!

Routing-Belastung eines Knotens:
Zahl von Routen, die durch den Knoten gehen

Routing-Belastung einer Verbindung:
Zahl von Routen, die durch die Verbindung gehen

Beispiel:

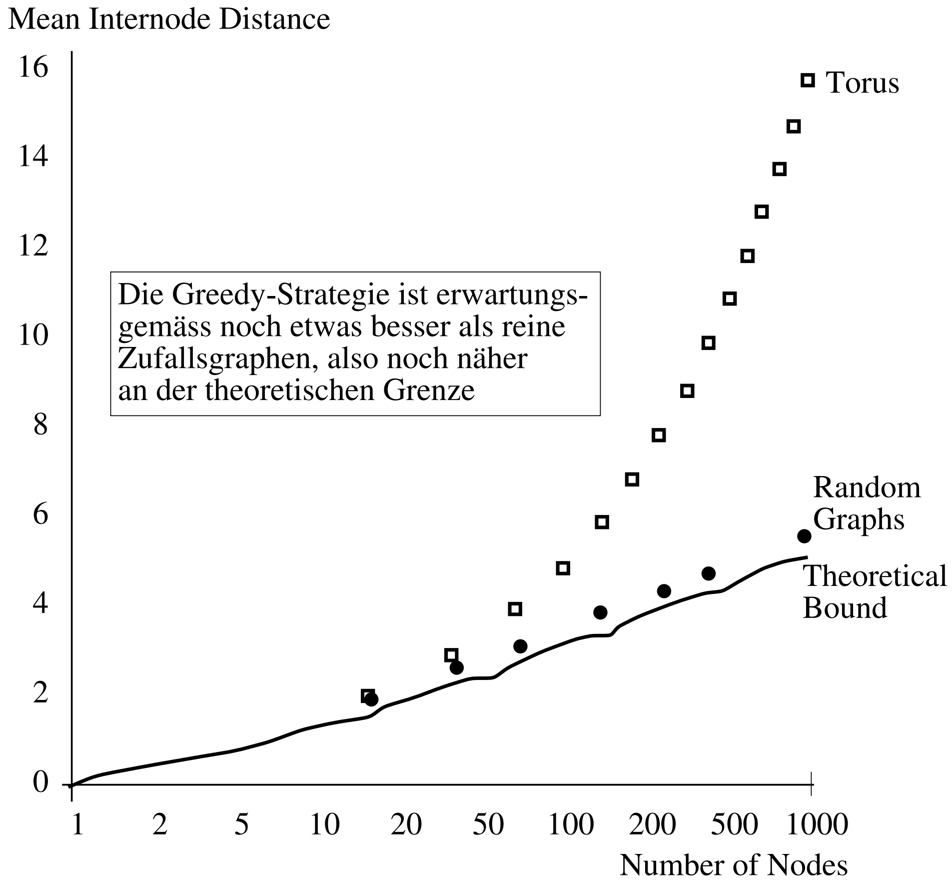


10 Routen von 20 gehen durch c

12 Routen von 20 gehen durch bc

Mittlerer Knotenabstand

- Für Knoten mit Grad 4
- Untersucht von D. Prior, ECSP (Edinburg)



Man lese zu den Untersuchungen zu Zufallstopologien folgenden Artikel:

D. M. N. Prior, M. G. Norman, N. J. Radcliffe, L. J. Clarke: *What Price Regularity*?, Concurrency, Practice and Experience, Vol 2 No 1, pp. 55-78, 1990

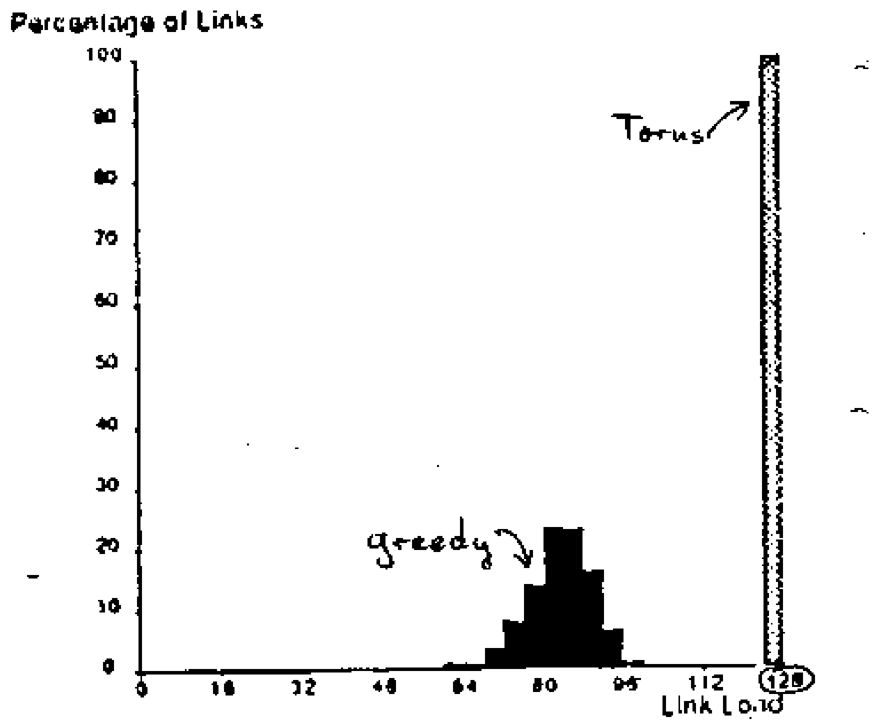
Link loads for 100 processors

- Wenn jeder mit jedem anderen kommuniziert: ca. 100^2 Nachrichten (auf kürzesten Pfaden)

- 10×10-Torus: mittlere Entfernung = 5

  --> Jede Nachricht benutzt im Mittel 5 links.

  Bei 100^2 Nachrichten, 400 links --> 125 Nachrichten / link

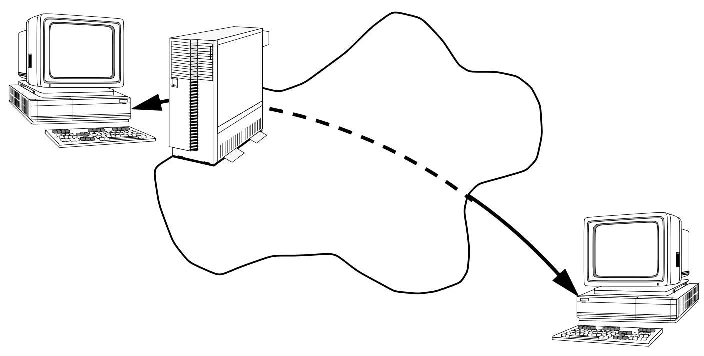


Rechnernetze

Rechnernetze

mehr dazu in speziellen Vorlesungen!

- Verbindung von autonomen Rechnern zum Zweck der Datenkommunikation



- Man unterscheidet:

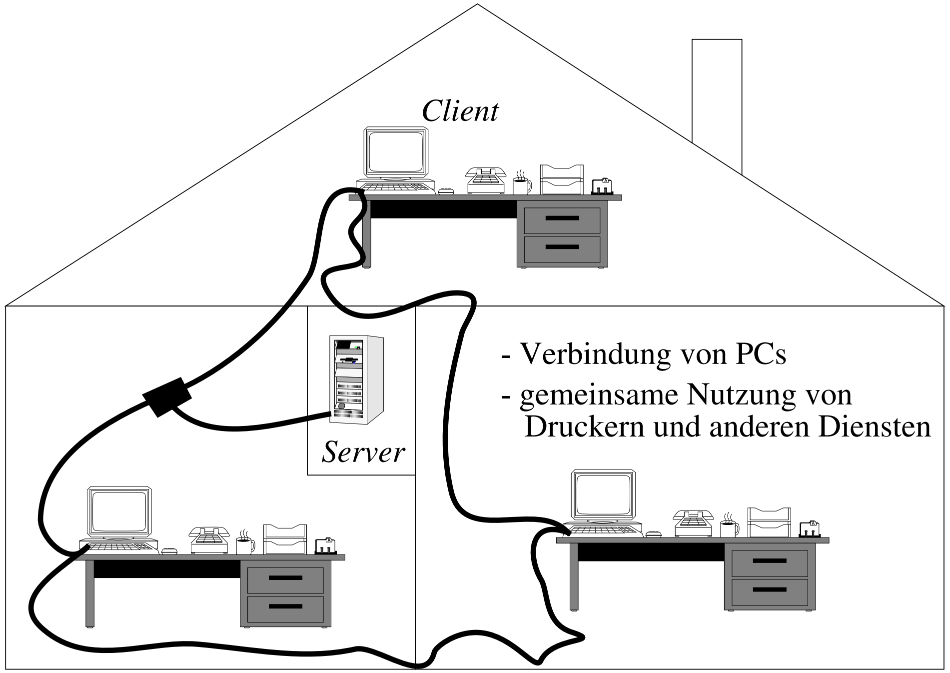
  - *Lokale Netze* (LAN, local area network):

    Rechner einer Abteilung oder einer Firma werden miteinander verbunden. Ausdehnung: einige 100 Meter.

  - *Weitverkehrsnetze* (WAN, long haul networks):

    Weltweite Verbindung über Kommunikationsnetze (spezielle Datennetze) bzw. private oder kommerzielle Satellitenverbindungen etc.

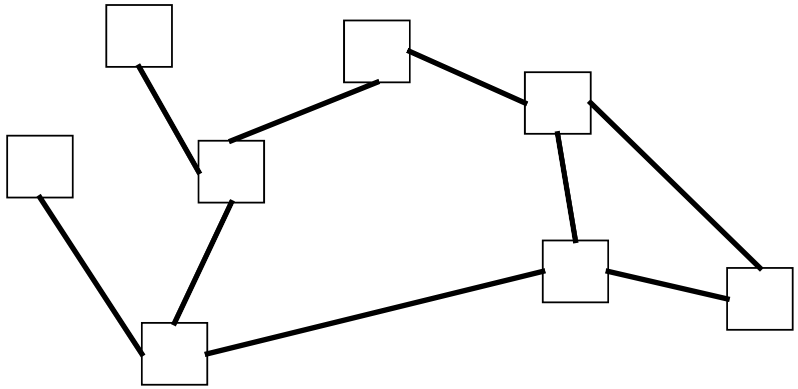
Lokale Rechnernetze



- Verbindung von PCs
- gemeinsame Nutzung von Druckern und anderen Diensten

- Ausdehnung wenige 100 Meter.
- Hohe Übertragungsrate (typ. 10 - 1000 M Bit / s).
- Typischerweise innerhalb eines Gebäudes.
- Oft: Möglichkeit zu Broadcast.
- Typisch: Client/Server-Modell
  - z.B. Clients als Arbeitsplatzrechner
  - Server als Massenspeicher, Mailserver, Datenbankserver...

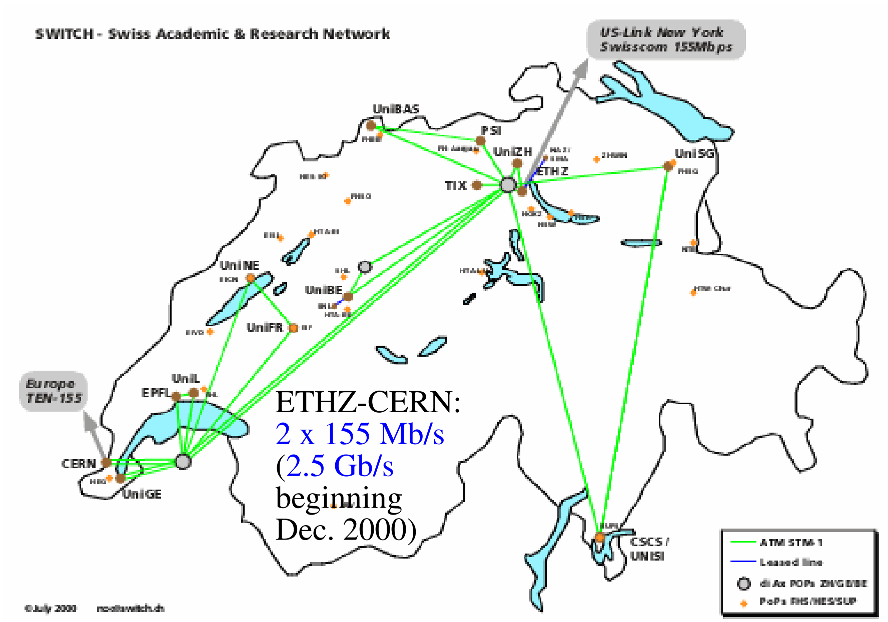
Weitverkehrsnetze



- Verbindungen (z.B. einige 100 oder 1000 km) über Standleitungen, Mikrowellen, Satelliten...
- Insgesamt schwach “vermascht” (Kosten!)
- Dezentral organisiert
- Nachrichten werden i.a. über andere Rechner geleitet (Routing; “store and forward”)
- Oft geringere Übertragungsrate als LANs
  - typisch: 50 k Bit/s (z.B. Modem) bis 2 M Bit/s (ggf. auch höher)

SWITCH (2000)

Swiss Academic & Research Network



http://www.switch.ch/lan/national.html

The SWITCHlan consists of an ATM core network which is built on top of diAx's national SDH infrastructure. STM-1 links are used to construct a double star ATM topology with hubs in Le Lignon (GE) and Altstetten (ZH). Every site on the ATM core is connected by two different STM-1 links which use physically diverse paths on the SDH rings. The local loops between the customer's site and the nearest diAx PoP are realized as dark fibers.

Access from ETHZ/SWITCH to WorldCom's Zurich PoP is realized through the City Ring Zurich. Two STM-1 links to Amsterdam and Frankfurt connect Switzerland to WorldCom'sATM Backbone.
From Switzerland to New York, the transatlantic connection uses two 155 Mb/s circuits which follow different paths through Europe and across the Atlantic ocean (geograpical diversity).The US-link from Zurich(ETHZ) via the AC-1 cable terminates at the SCNA PoP in New York, 111, 8th Avenue at a router owned by SWITCH. The link from Geneva (CERN) via the TAT-12 cable terminates at another SCNA PoP in New York, 60, Hudson Street, again at a router of SWITCH.

Beispiel: TAT-14-Transatlantik-Kabel

A consortium of telecom operators on Wednesday signed a **$1.5 billion** project for a new fibre-optic cable link between Europe and the U.S. in order to cut waiting time on the World Wide Web.

More than 50 telecommunications operators signed contracts for the **TAT-14** cable which will have a capacity of **640 Gb/s** and could carry about 7.7 million simultaneous telephone calls.

Some **80 percent** of its capacity will be allocated to **Internet** and multi-media traffic.

The TAT-14 cable network is a 1997 initiative of **11 carriers** - AT&T, BT, Cable & Wireless, Deutsche Telekom, France Telecom, KPN MCII, Pacific gateway Exchange, Sprint, Swisscom and Telia.

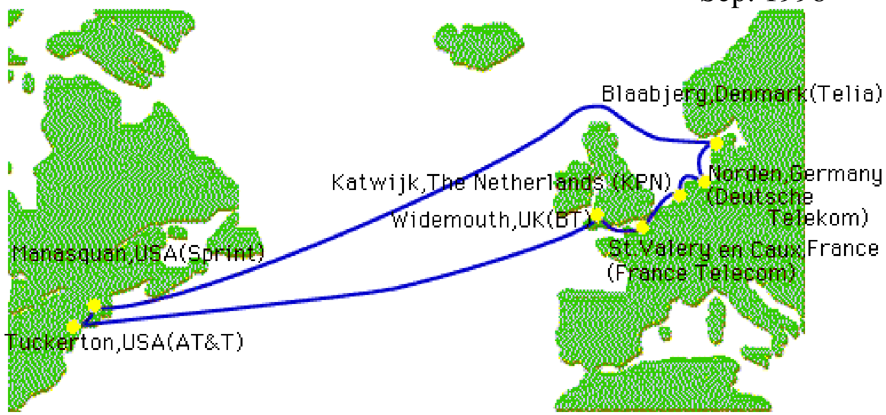
The TAT-14 cable will link five European countries - Germany, England, Denmark, France and the Netherlands - with the United States.

It will span 15,000 kilometres and is expected to be completed and in service by the **end of 2000**.

The new system, which is a ring network, will utilise the latest advances in Synchronous Digital Hierarchy (SDH), Wavelength Division Multi-plexing (WDM; 16 wavelengths of 9.953 Gb/s) technology and will con-sist of **four pairs of optical fibre cable**.

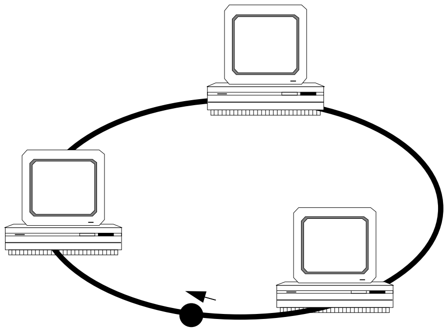
The new system will represent **64 times the capacity of the original TAT-12/TAT-13** cable network which was put into service in September 1996.

-- Sep. 1998



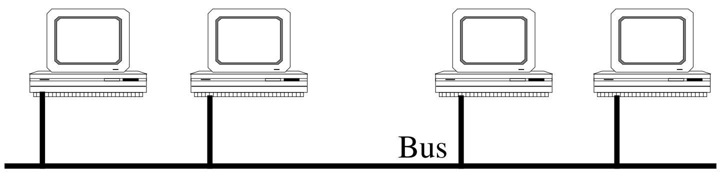
Topologien lokaler Rechnernetze

- *Token-Ring*



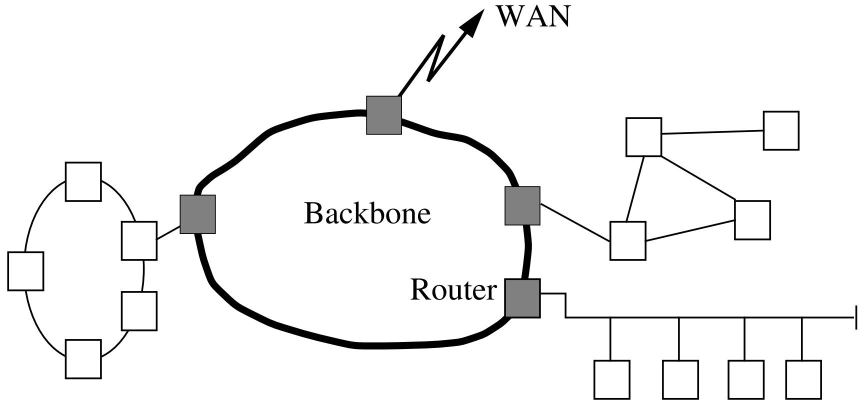
- Nachricht wird solange weitergereicht, bis die Zieladresse mit der angegebenen übereinstimmt.
- Es kreist ein sogenanntes “Token” (spezielles Paket).
- Ein Rechner darf nur dann senden, wenn er das Token besitzt.

- *Bus-Topologie* (z.B. *Ethernet*)



- Jeder Rechner “horcht” am Bus und empfängt die Nachrichten, die seine Adresse tragen.
- Senden, wenn Bus frei.
- Kollisionen von mehreren Sendern sind ein Problem.
- CSMA/CD (Carrier Sense Multiple Access / Collision Detection)

Backbone-Netze



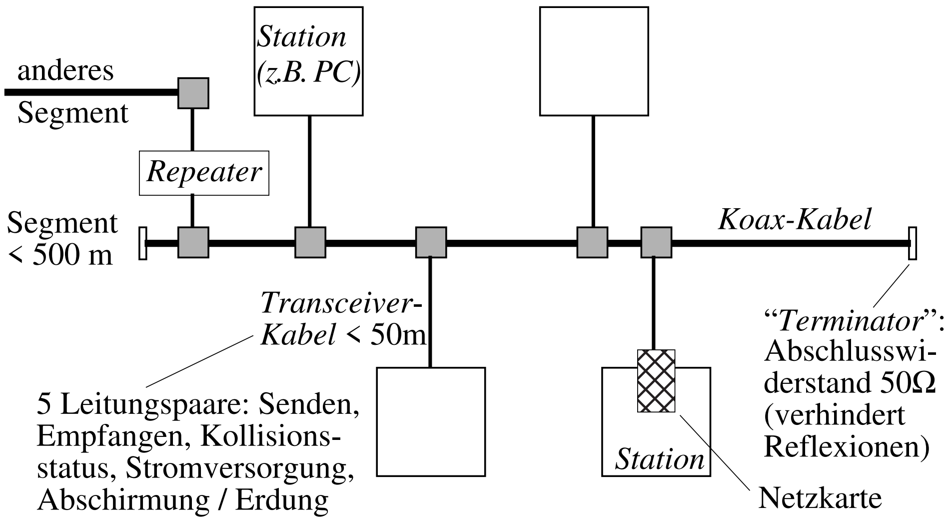
- Dient (nur) dazu, andere Netze (LANs) zu verbinden
- Leistungsfähiger als normale LANs
  - z.B. 1 Gb/s Ethernet oder ATM-Verbindung bei 10 Mb/s in den angeschlossenen LANs
- Räumliche Ausdehnung typischerweise Campus, Firma, grosses Gebäude...
- Unter “Internet-Backbone” werden “lange” Leitungen hoher Bandbreite (i.a. Glasfaser) zusammen mit Hochleistungsswitches und Routern etc. verstanden

Ethernet

"Äther"
--> gemeinsames Medium
--> broadcast der Nachrichten

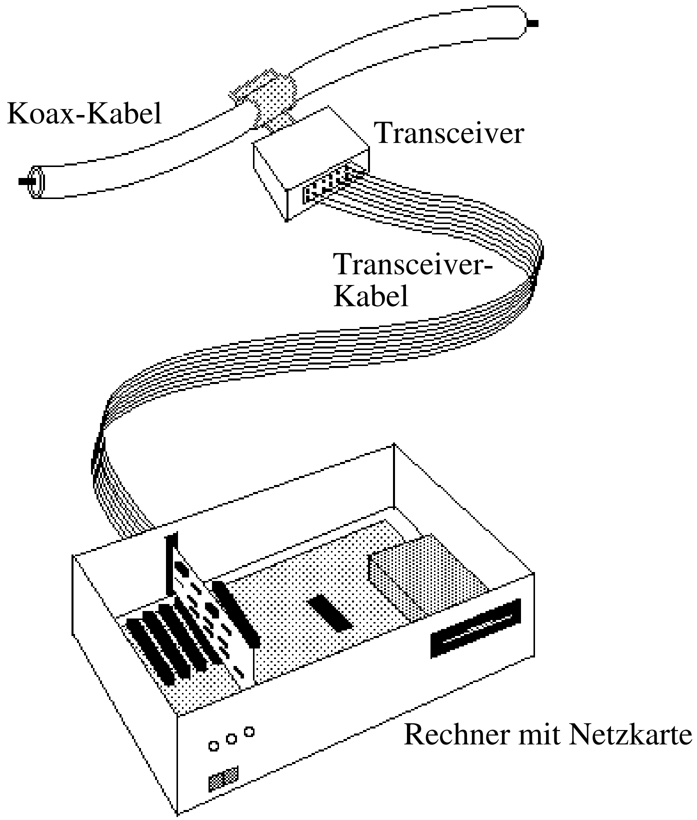
- Bus mit dezentraler Zugangskontrolle
- Industriestandard seit 1980
- Genormt durch IEEE 802.3
- Relativ unempfindlich gegenüber Ausfall einer Station
- Preiswert, einfach ausbaufähig, wenig Administration

Klassische Konfiguration:

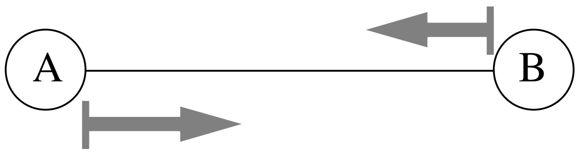


- Verbinden zweier Segmente durch *Repeater* (bidirektionale Signalverstärker) möglich
- Gesamtausdehnung max. ca. 2500 m

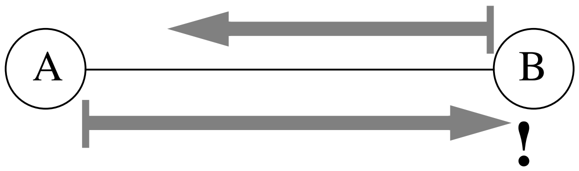
Klassisches Ethernet: Anschluss einer Station



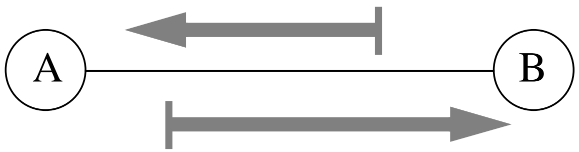
Kollisionen



A und B fangen "fast" gleichzeitig an zu senden.



Etwas später merkt B, dass A auch sendet.

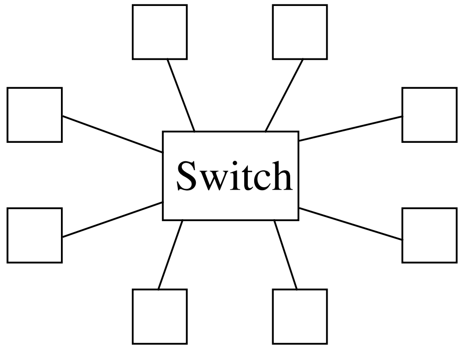


Das darf <u>nicht</u> auftreten: Keine Station merkt (während sie sendet!), dass eine andere ebenfalls sendet.

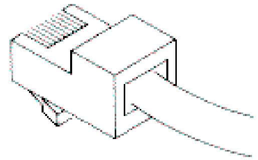
==> - Maximalentfernung zweier Stationen
- Mindestlänge eines Nachrichtenpakets (--> Mindestdauer eines Sendevorgangs)

Twisted-Pair-Ethernet

- "10 Base T"; i.a. sternförmig



RJ-45-Anschluss:



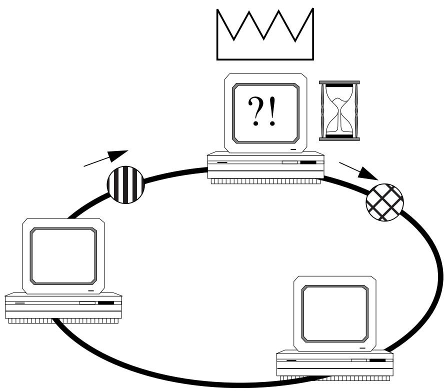
- Kabelstörungen betreffen i.a. nur eine einzige Station
  - Switch entfernt im Fehlerfall betroffene Station logisch
- "Switch"
  - stellt jedem Endgerät die volle Bandbreite zur Verfügung
  - schaltet dedizierte Verbindungen zwischen je zwei Ports
  - simuliert auf jeder Leitung eigenes Ethernet (--> weniger Kollisionen)
  - bei einem belegten Zielport muss ein Paket ggf. (kurz!) gepuffert werden, oder es wird dem Sender ein Kollisionssignal geschickt

Token-Ring: der Monitor

- Eine Station übernimmt die Rolle des *Monitors*.

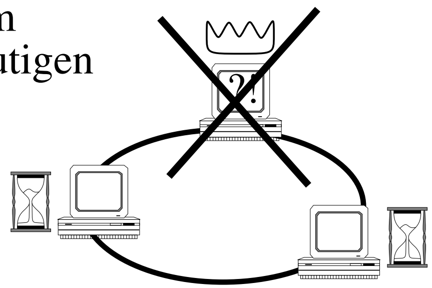
  --> Erkennung und Behebung von Fehlersituationen

  - Timeout, um verlorene Token zu erkennen (--> regenerieren)
  - Stempelt Nachrichten, um mehrfaches Zirkulieren zu erkennen.
  - Absorbiert alte Token und fehlerhafte Nachrichtenreste.



- *Monitorausfall* wird von anderen Stationen durch einen "grossen" Timeout (kein Token gesehen) erkannt.

  --> Election-Protokoll, um *dezentral* einen eindeutigen Monitor zu wählen.

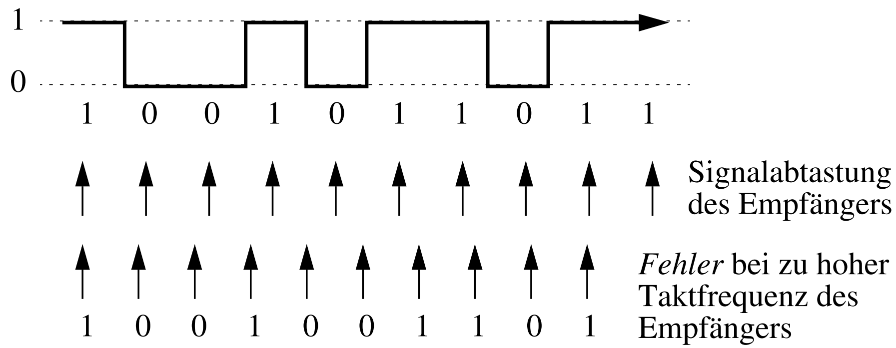


Election-Protokoll wird auch benötigt, damit die Stationen nach "erstmaliger" Konfiguration einen Monitor auswählen.

Taktsynchronisation; Datenpakete

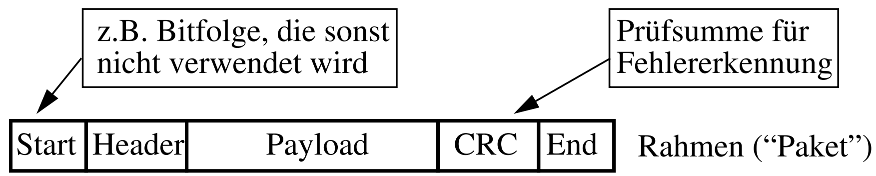
-"Gleichlauf" von Sender und Empfänger notwendig

- *Taktsynchronisation* notwendig zur korrekten Interpretation übertragener Signale (Zeitpunkt zum Abtasten des Signals)



- hierzu verschiedene Techniken: Taktsignal über getrennte Leitung oder Regenerierung des Taktes aus dem Übertragungssignal (dann genügend viele Wertänderungen im Signalstrom notwendig!) und punktuelle Resynchronisation eines unabhängigen Taktgenerators

- Typischerweise werden mehrere Zeichen zu einem Datenpaket verbunden und als Einheit übertragen
  - als Einheit zur Fehlerprüfung, Quittierung, Wiederholung etc.



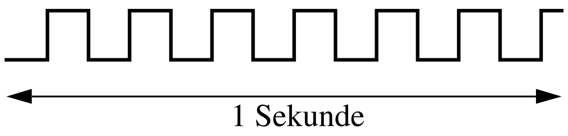
- Länge von Datenpaketen z.B. 53 Byte bei ATM, 64-1500 Byte bei Ethernet, max. 64 kByte bei IP

- Header enthält typischerweise die Länge des Pakets, Adressinformation und Steuerinformation (Flags etc.)

Bandbreite und Delay

- *Bandbreite*: Übertragene Datenmenge pro Zeiteinheit
    - Beispiel Ethernet: 10 Mb/s ("klassisch") bzw. 100 Mb/s ("fast")

- Bandbreite veranschaulicht durch "Bitbreite":

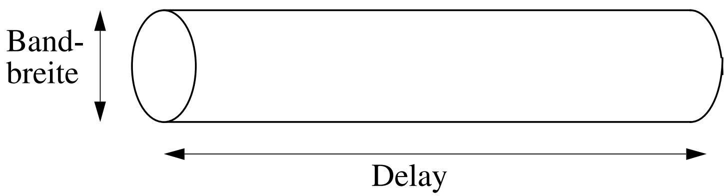


1 Mb/s
(jedes Bit eine Mikrosekunde breit)

2 Mb/s
(jedes Bit 0.5 Mikrosekunden breit)

- *Delay*: Zeit, um eine Nachricht von A nach B zu senden
    - Für kurze Nachrichten ("Null-Message") ist Delay entscheidend; Bandbreite ist vernachlässigbar.
    - Für grossvolumige Daten (z.B. Bilder) ist die Bandbreite der dominierende Faktor.

- Bandbreite-Delay-Produkt oft interessant:



- Beispiel: 100 ms Delay und 45 Mb/s Bandbreite
  --> 562 KB Daten, die unterwegs sind, bevor erstes Bit ankommt
- Speichervermögen eines Kommunikationskanals

Hochgeschwindigkeitsnetze

ca. 100 Mbit/s - 1 Gbit/s

- "Fast Ethernet" (100 Mb/s) in LANs
    - mit Twisted-pair-Kabel und i.a. über Switch (sternförmig)
    - Switch hat i.a. Ports, die sowohl 10 Mb/s als auch 100 Mb/s können

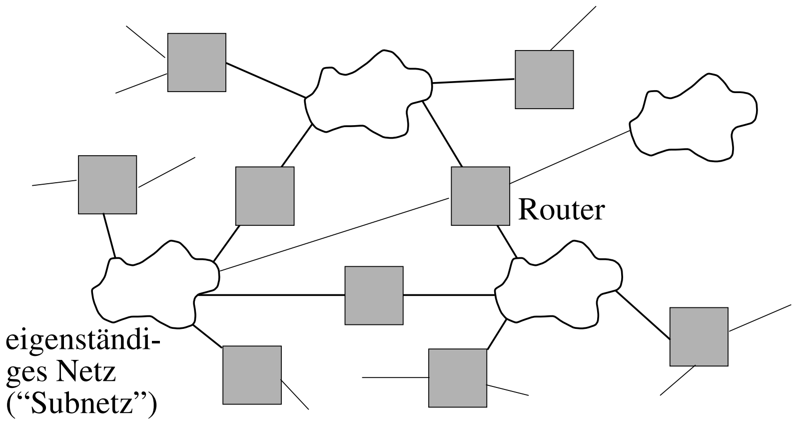
- "Gigabit-Ethernet" (1 000 000 000 b/s) in LANs
    - über Switch (sternförmig) und typw. Glasfaser
    - z.Z. noch wenig verbreitet (neu, teuer)

- ATM (Asynchronous Transfer Mode)
    - für WAN und LAN (in LANs aber teurer als Ethernet)
    - Grundlage für Breitband-ISDN
    - Pakete ("Zellen") von konstant 48 Byte mit 5 Byte Header
    - kleine Zellen: Zelle ist schnell voll und kann gleich verschickt werden
    - typische Bandbreiten: 155 Mb/s; ca. 600 Mb/s
    - für Audio, Video und Realzeitdaten gut geeignet ("Dienstgüte")

- Engpass ist oft nicht mehr die "Leitung", sondern die Geschwindigkeit der Kommunikationssoftware

- "Hochgeschwindigkeit" flächendeckend in die Haushalte?
    - Kabel-TV-Infrastruktur aufrüsten ("Kabelmodem" beim Kunden und Anbindung des "Headends" über Glasfaser an die Internet-Infrastruktur)
    - Telefon-Infrastruktur aufrüsten (z.B.: xDSL mit ca. 1-8 Mb/s)
    - Glasfaser in die Haushalte (teuer)
    - Wireless Local Loop (WLL)
    - Powerline: Daten huckepack über das 230V-Netz
    - noch Science Fiction: tieffliegende Satelliten, spezielle Flugzeuge etc.

Inter-Netzwerke



- Koppelung von LANs, Intranets, Campusnetze, Firmennetze etc. über "Gateways"
  - ggf. mit Protokollumsetzung oder -anpassung

- "Internet": Verwendung einheitlicher Protokolle
  - basierend auf IP = Internet Protocol
  - Abschottung lokaler Netze durch Firewalls

- Routing ("Leitwegbestimmung")
  - Ermittlung von Routing-Tabellen
  - Weiterleitung einer Nachricht mittels Routing-Tabellen ("Forwarding")
  - es gibt diverse Routing-Algorithmen (oft: dezentral)

Kommunikationsprotokolle

- Kommunikation in verteilten Systemen geschieht ausschliesslich über *Nachrichten*.

- Hierbei sind gewisse *Konventionen* und *Normen* zu beachten, damit die Kommunikation klappt.

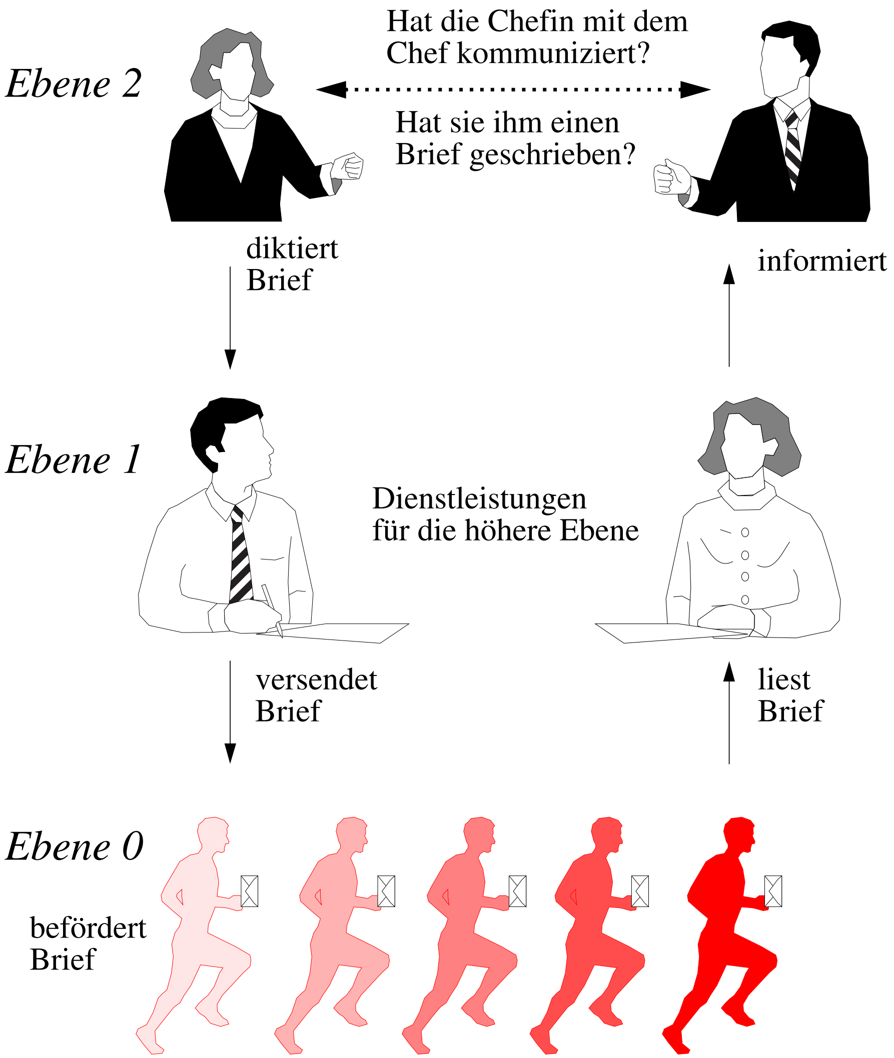
- *Protokoll* = Festlegung der Regeln und des algorithmischen Ablaufs bei der Kommunikation zwischen 2 oder mehr Partnern.

- Es müssen viele Vereinbarungen getroffen werden, z.B:
  - Steckergrösse
  - Wieviel Volt repräsentieren "0" bzw. "1"?
  - Ist das erste Bit vorne oder hinten?
  - Was tun bei einer fehlerhaften Übertragung?
  - Wird ein Rasterbild zeilen- oder spaltenweise übertragen?
  - ...

--> Vereinbarungen auf z.T. unabhängigen "Ebenen"

--> Bildung sinnvoller Schichten

Schichten ("layers") und Ebenen



Schichten (2)

- Eigentliche ("physische") Kommunikation geschieht auf der untersten Ebene

- Auch mittlere Ebenen (Sachbearbeiter) kommunizieren miteinander

- Nehmen dazu *lokale Dienste* einer tieferen Ebene in Anspruch

- Ebene 0 kann ausgetauscht werden (z.B. Fax statt gelbe Post), ohne dass sich auf höherer Ebene etwas ändern muss (--> Transparenz)

- Ebenen 0 und 1 können ausgetauscht werden (z.B. wenn Ebene 2 den Telefondienst statt dessen in Anspruch nimmt)

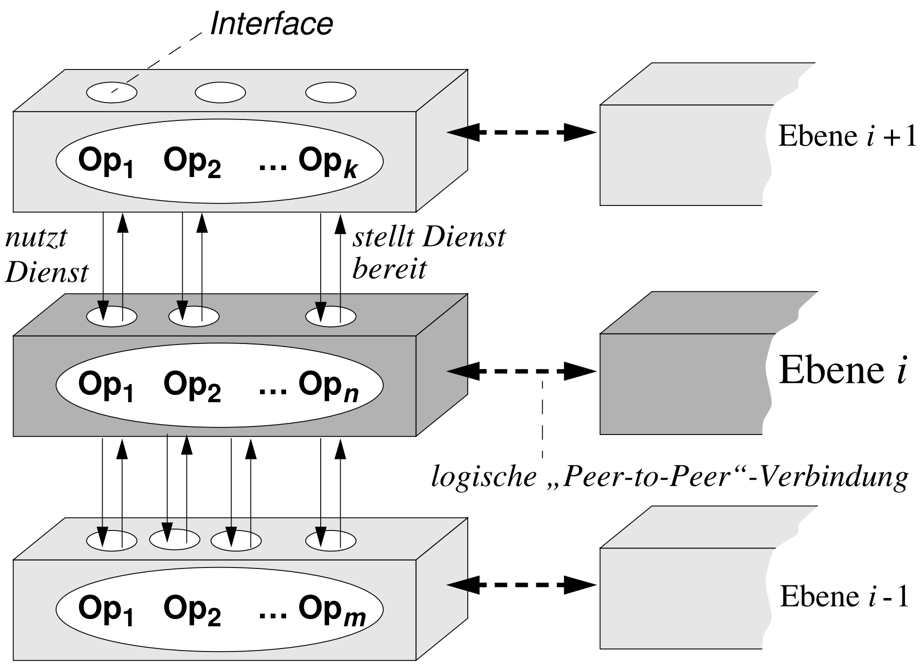
Vgl. auch Speditionsfirma, die verscheidene Transportdienste in Anspruch nehmen kann:

- Auf jeder Ebene gelten jeweils eigene Regeln (StVO...: Rechtsverkehr...), die aber nicht nach oben durchschlagen

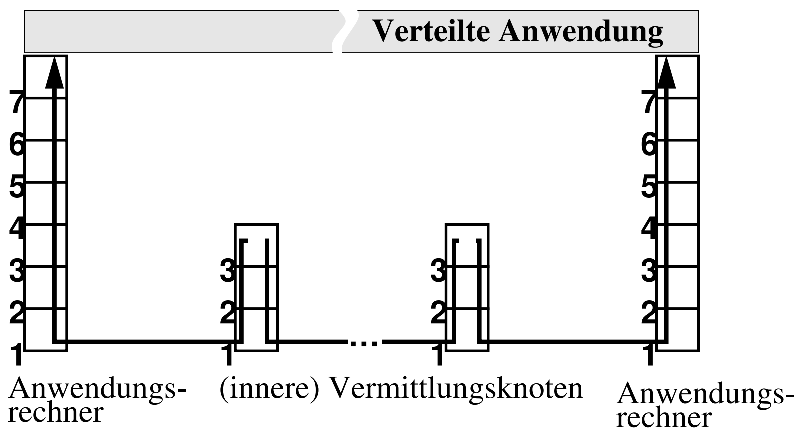
- Leistungsfähigkeit eines Dienstes (Kosten, Geschwindigkeit, Güte...) sind allerdings spürbar

- Verschiedene Transportdienste bieten i.w. den gleichen Service --> sind austauschbar

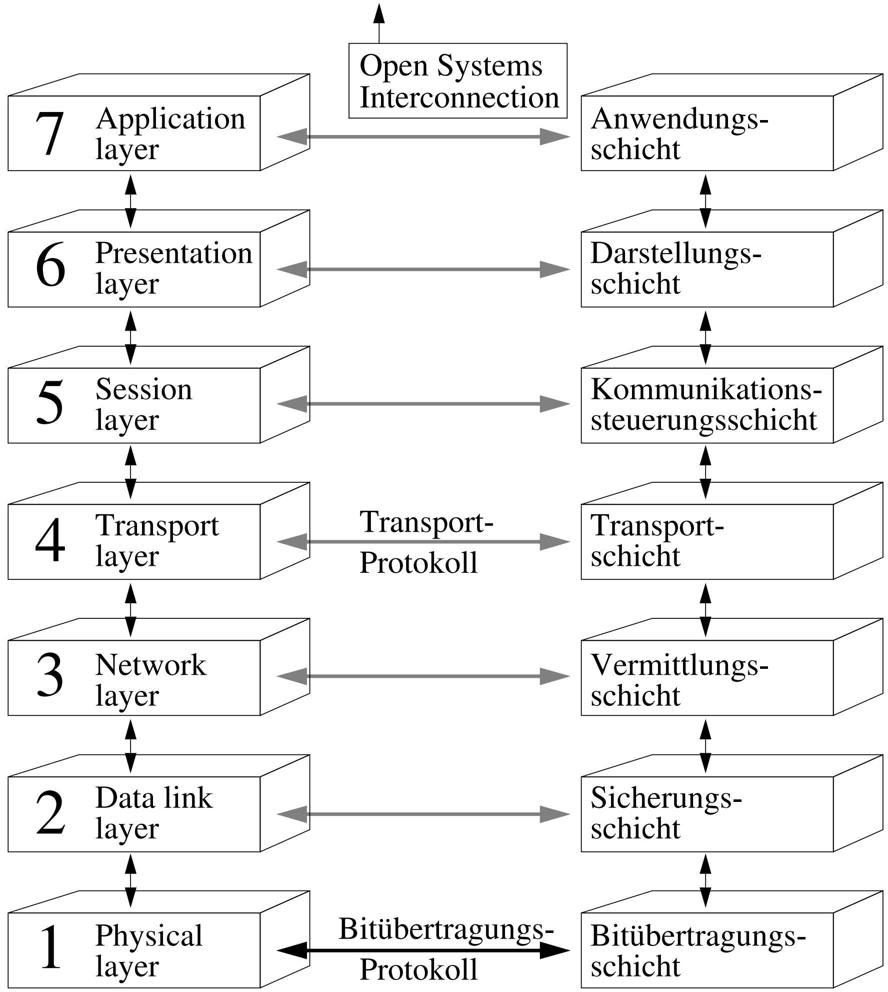
Prinzip der Protokollschichtung



- Transport von Nachrichten über Vermittlungsknoten:



Das ISO-OSI-Referenzmodell



- Beachte: Dies ist ein “Referenzmodell”!
  - Strukturelle Basis (“Architekturmodell”) für Protokolle und deren Normierung (einheitliches Vokabular; Orientierung etc.)

- Wieso gerade 7 Schichten?
  - In einigen Protokollen sind einige Schichten fast “leer”
  - In einigen Protokollen werden einige Schichten nochmals unterteilt

Wichtigste Aufgaben der 7 Schichten

1. *Physical Layer*
   - Normung von Steckern und Kabeleigenschaften
   - Kodierung von Bitfolgen mit physikalischen Signalen

2. *Data Link Layer*
   - Flusssteuerung ("flow control"): Sender ggf. bremsen
   - Fehlerfreie Übertragung von Bit Frames zwischen zwei Nachbarknoten

3. *Network Layer*
   - Übertragung von Paketen zwischen beliebigen Endknoten
   - Routing
   - Multiplexen von Verbindungen

4. *Transport Layer*
   - Logische Verbindung zwischen (adressierten!) Prozessen (bzw. ports, sockets...) statt Rechnern
   - Bitströme bzw. Nachrichten beliebiger Länge werden in Pakete aufgeteilt

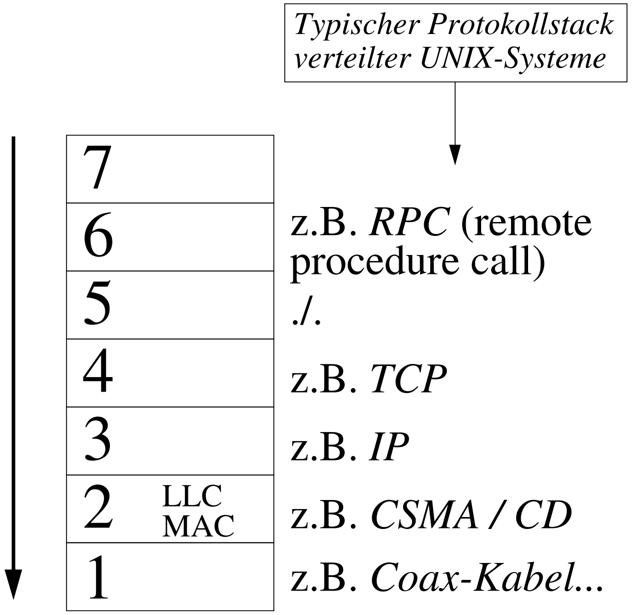
5. *Session Layer*
   - Sitzungsverwaltung über Phasen (z.B. Login / Logout) hinweg
   - Kopplung mehrerer Transportverbindungen (z.B. Audio + Video) zu einer Sitzung

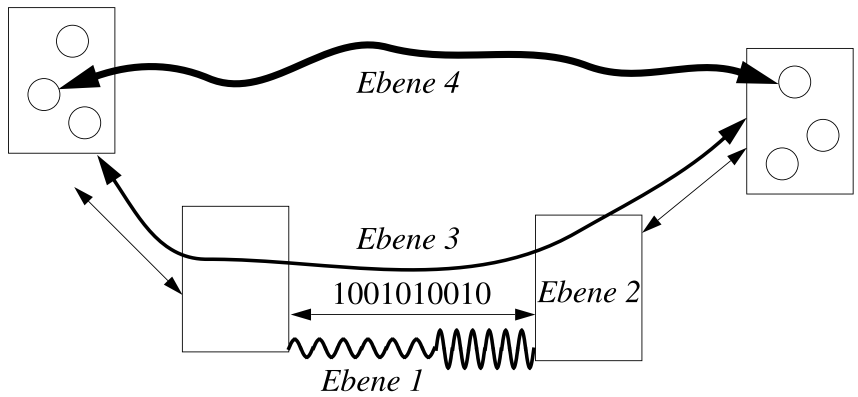
6. *Presentation Layer*
   - Kodierung komplexer Daten (Typ, Wertebereich, Struktur...)

7. *Application Layer*
   - z.B. WWW (http-Protokoll) oder ftp

Protokoll-Stack

- Menge der in einem gegebenen Fall verwendeten spezifischen Protokolle





Die Internet-Protokollhierarchie

	Application Layer		Gegenüber OSI sind die drei obersten Schichten zusammengefasst
4	Transport Layer		TCP (bzw. UDP)
3	Internet Layer		Verfahren, um mehrere verschiedene Netze zu traversieren; Wegewahl; IP
			In LANs Unterteilung in zwei Teilschichten:
2	Network Access Layer	Logical Link Control (LLC)	Fehlerfreie Übertragung;
		Medium Access Control (MAC)	Zugriff auf das Medium
1	Physical Layer (PHY)		Bitübertragung

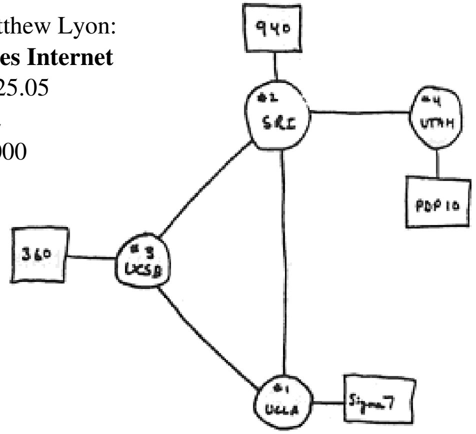
- Im Internet gehört alles oberhalb der TCP-Ebene zur "Anwendungsebene"
  - Dienstprogramme wie dns oder ftp
  - Echte verteilte Anwendungen
  - "Middleware", die eine eigene Infrastruktur für grosse verteilte Anwendungssysteme bereitstellt

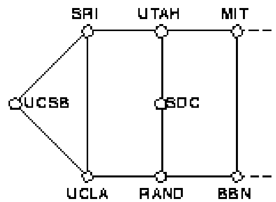
Internet-Geschichte: Pionierzeit

- 1967: ARPA (Advanced Research Project Agency) des DoD vergibt Auftrag "Projektstudie ausfalltolerantes Paketnetz" an SRI (Stanford Research Institute)

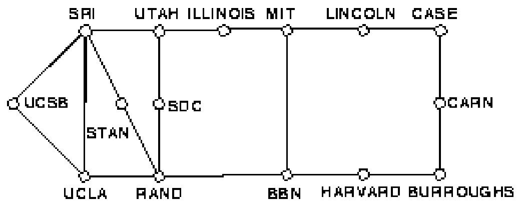
- 1969: Erstes "Internet" aus 4 Knoten: University of California at Los Angeles (UCLA), Stanford Research Institute (SRI), University of California at Santa Barbara (UCSB), University of Utah

Katie Hafner, Matthew Lyon:
Die Geschichte des Internet
351 Seiten, EUR 25.05
ISBN: 932588592
dpunkt-Verlag, 2000





Juli 1970



März 1971

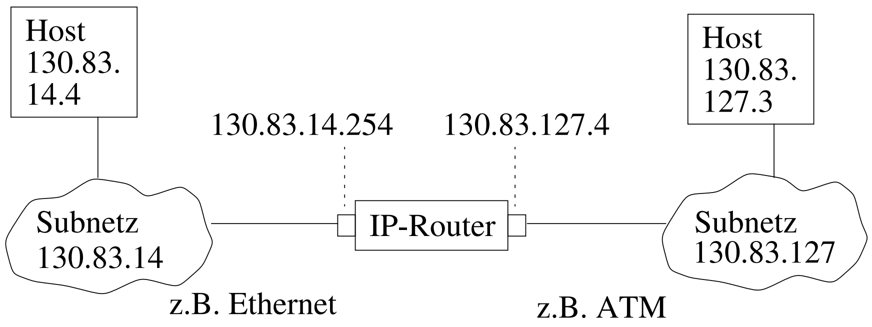
- 1971: Betriebsaufnahme des ARPAnet; Experiment zum Einloggen auf entfernten Rechnern; erstmalig Nutzung von E-Mail
- 1972: Erste öffentliche Demonstration des Netzes
- 1974: Neue Protokollsuite: TCP/IP
- 1980: Integration der TCP/TP-Protokolle in UNIX

Das Internet

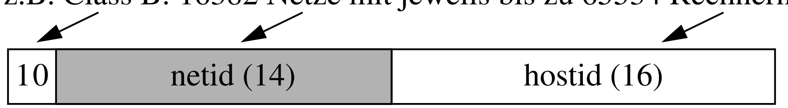
- Geschichte
  - Pionierzeit (USA): 70er Jahre des 20. Jahrhunderts
  - Wissenschaftsnetz (getrieben durch die Informatik): 80er Jahre
  - Kommerzialisierung und Popularisierung: 90er Jahre

- Technologie
  - globales Addressierungsschema ("IP-Adressen")
  - TCP/IP-Protokolle + Routing-Verfahren + Hilfsdienste
  - Anwendungsprotokolle: E-mail, ftp, news, telnet, http,...

- Jede Maschine am Internet hat eine IP-Adresse
  - 32 Bit lang (zukünftige IPv6-Version: 128 Bit)
  - typischerweise als 4 Dezimalzahlen geschrieben
  - Bsp.: 192.130.10.121 (= 11000000.10000010.00001010.01111001)



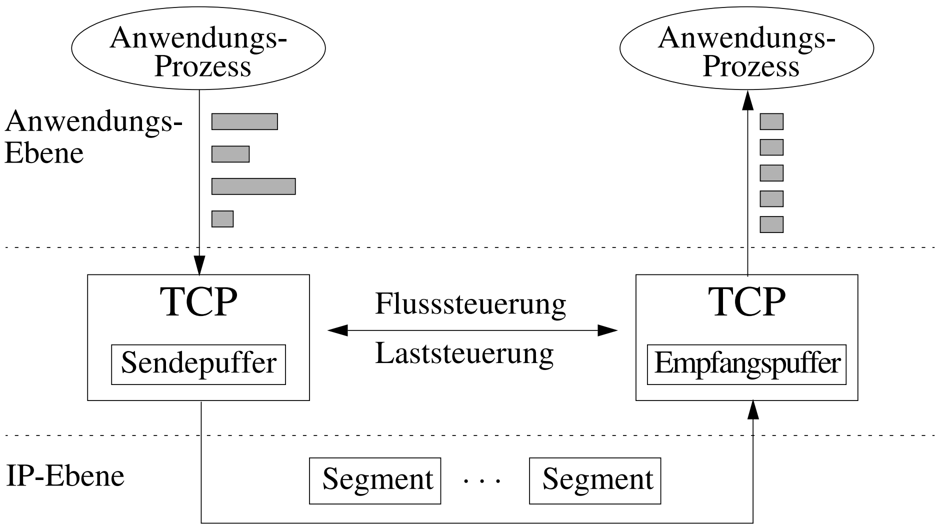
- Adressen werden in einen Netz- und Host-Teil aufgespalten
  - z.B. Class B: 16382 Netze mit jeweils bis zu 65534 Rechnern



TCP (Transmission Control Protocol)

- TCP bietet einen zuverlässigen Kanal für einen Bytestrom
  - zwischen zwei Ports (=Kommunikationsendpunkte für Programme)
  - in beide Richtungen ("voll duplex")
  - unstrukturiert (nur Bytes; Nachrichtengrenzen gehen verloren)

- Verbindungsaufbau vor Datenübertragung notwendig
  - Verbindungsabbau danach



- Ein spezielles Protokoll ("sliding window") sorgt für eine gegen Übertragungsfehler gesicherte Kommunikation sowie die Fluss- und Laststeuerung
  - *Flusssteuerung*: Sender soll *Empfänger* nicht überlasten
  - *Laststeuerung* : Sender soll das *Netz* nicht verstopfen