Resümee (1)

- Kooperation durch Kommunikation

- Verteilte Systeme
  - keine globale Sicht
  - keine gemeinsame Zeit
  - parallel
  - nicht-deterministisch
  - unbestimmte Nachrichtenlaufzeit

- Typische Probleme verteilter Systeme / Algorithmen:
  - Beobachtungsproblem (keine Gleichzeitigkeit)
  - Schnappschussproblem (wieviel Geld ist in Umlauf?)
  - Terminierungserkennungsproblem
  - Deadlockproblem (Phantomdeadlock?)
  - Kausalitätsproblem (indirekte Wirkung vor Ursache)

- Problem globaler Prädikate ("relativistischer Effekt")
  - es gibt i.Allg. mehrere "gleichberechtigte" Beobachter
  - diese stimmen i.a. bzgl. der Gültigkeit des Prädikates nicht überein!
  - gibt es beobachterinvariante Prädikate?

- Verteilter Euklidischer Algorithmus
  - als erstes Beispiel für einen verteilten Algorithmus
  - reaktives Verhalten ("nachrichtengesteuert")
  - Korrektheit der Idee / des konkreten Algorithmus? (Invarianten...)

Resümee (2)

- Konzeptuelle Hilfsmittel
  - Zeitdiagramme
  - Atomare Aktionen
- Vorüberlegungen zu Übungen 1
  - potentielle Kauslität als Halbordnung über "Ereignissen"
  - kausaltreue Beobachtung
- Flooding-Algorithmus
  - Nachrichtenzahl
  - Problem der Terminierungserkennung
  - Formalere Fassung in Pseudo-Code
- Echo-Algorithmus (Variante von Flooding)
  - Nachrichtenzahl 2e
  - Explorer- / Echo-Welle
  - Spannbaum
  - Zwei "disjunkte" Wellen (rot; grün)
  - Verbesserung durch Mitführen von Knotenidentitäten?

Resümee (3)

- Zeitkomplexität
  - Einheitszeitkomplexität
  - Variable Zeitkomplexität

- Broadcast auf Ring

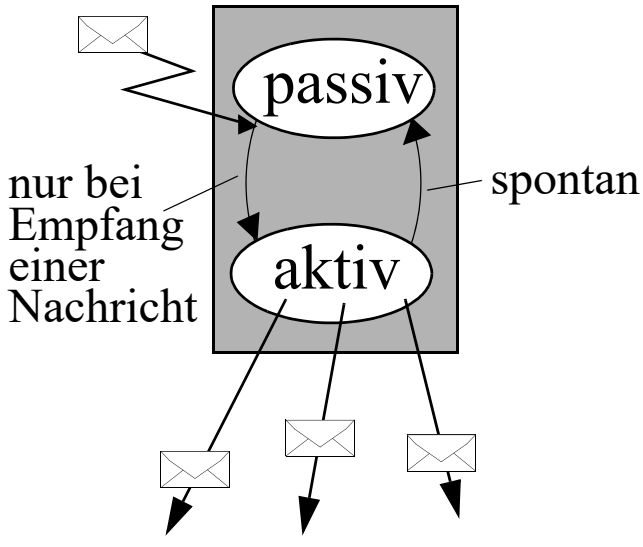
- Broadcasts auf Hypercubes
  - Hypercube: Definition und Eigenschaften
  - Einzelnachrichten: Routingverfahren
  - Broadcast entsprechend der rekursiven Definition
  - Broadcast durch Fluten in jeweils höhere Dimensionen
  - Optimalität (Nachrichten- und Zeitkomplexität) des Broadcastproblems
  - schneller Broadcast durch paralleles Senden von Teilnachrichten

- Berechnung von Routingtabellen
  - verteilte Version des Bellmann-Ford-Algorithmus
  - auch wieder das bekannte Schema der verteilten Approximation
  - Anwendung in Rechnernetzen

- Paradigma der verteilten Approximation
  - Verallgemeinerung verschiedener ähnlicher Algorithmen

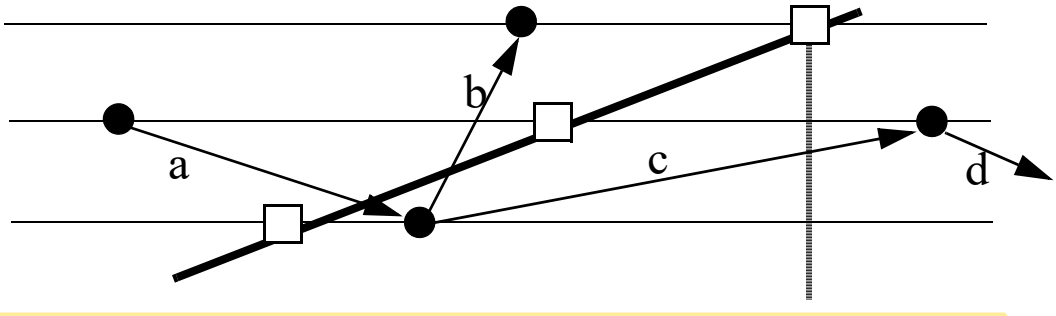
Resümee (4)

- Verteilte Terminierung
  - Problemdefinition



- Atommodell
  - Vereinfacht die Betrachtung des Wesentlichen
  - Terminierungskriterium: "keine Nachricht unterwegs"

- Schiefes Bild beim Beobachten verteilter Berechnungen
  - → Pauschales Zählen von Nachrichten genügt nicht
  - → Suche nach den eigentlichen Ursachen für Fehlschlag des Zählkriteriums ("Kompensation" der Zähler")



- Lösungsansätze zur Terminierungserkennung
  - Durch Vermeidung der "Ursachen" für das schiefe Bild
  - *Eindeutige Nachrichtennamen*
  - *Kanalzählerkriterium* (Widerspruchsbeweis: es gibt kein frühestes Ereignis nach dem Schnitt)

- Terminierungserkennung: *Doppelzählverfahren*
  - informeller Beweis (Aussage über gedachten senkrechten Schnitt zwischen den beiden Wellen)

- Kontrolltopologien zur Realisierung von Schnitten
  - Ring
  - Spannbaum

Resümee (5a)

- Safety- und Liveness-Eigenschaften verteilter Algorithmen
- Terminierungserkennung: *Zeitzonenverfahren*
  - Prinzip: *Erkenne* "Nachricht aus der Zukunft"
  - binäre "schwarz/weiss"-Zeit genügt
- *Einfrieren*?
- Terminierungserkennung: Vermeiden inkonsistenter Schnitte durch geeignetes *Vorziehen der Schnittlinie*

- Besprechung Teile von Übung 1:
  - Formalisierung von Zeitdiagrammen und "kausal abhängig"
  - kausaltreue Beobachtungen als lineare Erweiterungen ("Einbettung") der halbgeordneten Kausalitätsrelation

---

- Synchrones / asynchrones Senden
  - synchron: senkrechte Nachrichtenpfeile sind gerechtfertigt
  - nicht alles geht synchron (z.B. Überholen von Nachrichten)

Resümee (5b)

- Charakterisierung synchroner Kommunikation, z.B.:
  - alle Nachrichtenpfeile können senkrecht gezeichnet werden; Kommunikationskanäle sind immer leer
  - es gibt eine lineare Erweiterung der Kausalitätsrelation, so dass ein Empfangsereignis immer direkt nach seinem Sendeereignis kommt
  - Senden und Empfangen bilden "atomare Einheit"
  - Zyklenfreiheit der "synchronen Kausalitätsrelation <<" ("common past" / " common future"); dadurch Identifizierung von send und receive

zusammengehörige send/receive-Ereignisse sind "in gewissem Sinne" atomar

- Fragen...
  - sind die Charakterisierungen alle äquivalent?
  - kann man nun Nachrichtenlaufzeiten immer vernachlässigen?
  - funktioniert ein Algorithmus, der unter der Voraussetzung synchroner Kommunikation gemacht wurde, auch bei asynchroner Kommunikation?
  - und umgekehrt?
  - Terminierungserkennung bei synchroner Kommunikation? (das Atommodell ist dann offenbar nicht mehr adäquat, oder?)

Resümee (6a)

- Def. verteilte Terminierung bei synchroner Kommunikation

X_p: {$state_p$ = *aktiv*} $state_q$:= *aktiv* // "atomares" Aktivieren

I_p: $state_p$:= *passiv*

- Verhaltensmodelle verteilter Anwendungen
  - Transaktionsmodell
  - Atommodell
  - Synchronmodell

  } gegenseitige Simulation bzw. Transformation der Modelle

- Terminierungserkennung bei synchroner Kommunikation
  - z.B. Erkennen einer senkrecht von oben nach unten laufenden Nachricht, die einen schrägen Schnitt ("Welle") überquert

- Algorithmus von Dijkstra et al. ("DFG")
  - Schwarz- / Weiss-Färbung; Token auf einem Kontrollring
  - Beschreibung durch Menge von Verhaltensregeln

- Parallele Berechnungsschemata
  - Bsp.: Integration mittels Trapezmethode
  - Lastausgleich durch Migration von Arbeitseinheiten
  - Gesamtlast = 0 $\Leftrightarrow$ Terminierung

Resümee (6b)

- Terminierungserkennung mit der Kreditmethode
  (Einsammeln von "Krümeln")

- Safety: "Gesamtkredit" ist invariant

- Realisierung in verschiedenen Varianten möglich:
  - geeignete Darstellung der Krümel (negativer Zweierlogarithmus)
  - geeignete Realisierung des Einsammelns (Liveness!)

- Kreditmethode: Nachhalten fehlender Kreditanteile mit binärem Subtraktionsalgorithmus
  - in expliziter Mengenschreibweise

- Variante: direktes Nachlaufen
  - Analogie zum Echo-Algorithmus ("diffusing computations")!
  - Basisnachricht entspricht *Explorer*
  - Terminierungserkennung mit *Echos*

- Besprechung von Teilen von Übung 3
  - falscher Terminierungserkennungsalgorithmus
  - es genügt nicht, nur über den Zustand seiner direkten Nachbarn informiert zu sein!

Resümee (7)

- Wechselseitiger Ausschluss
  - safety
  - liveness
  - fairness

- Maekawa's $O(\sqrt{n})$-Algorithmus (Request-basiert)
  - Prinzip: Gitteranordnung; Request-granting-Mengen
  - Mimimale Mengen mittels endlichen projektiven Ebenen
  - $\sqrt{2}\sqrt{n}$ Dreiecksanordnung

- Wechselseitiger Ausschluss: Token-basierte Lösungen
  - kreisendes Token auf Ring

- Algorithmus von Ricart / Agrawala
  - Anforderungsnachrichten enthalten Zeitstempel
  - Token hat Auftragsliste und merkt sich Zeitpunkt des letzten Besuchs für alle besuchten Prozesse

- "Lift"-Algorithmus für wechselseitigen Ausschluss
  - Spannbaum / Baum: Umdrehen durchlaufener Kanten ("path reversal") beim Zurückholen des Tokens
  - O(log n) bei "guten" Bäumen
  - Verallgemeinerung auf beliebige (gerichtete azykl.) Graphen
  - Invarianten: Zyklenfreiheit; alle Pfade führen zum Tokenbesitzer
  - Request holt Token stets ein

- Vergleich von Algorithmen für den wechselseitigen Ausschluss (quantitative und qualitative Kriterien)

Resümee (8)

- *Election-Problem*: Symmetriebrechung
  - Auswahl genau eines Prozesses aus mehreren gleichartigen
  - falls Knoten nummeriert: Bestimmung des maximalen Knotens
- Election-Algorithmus mit dem Message-Extinction-Prinzip
  - verteiltes Approximationsschema
  - funktioniert auf allgemeinen (zusammenhängenden) Graphen
  - aber: Problem der Terminierungserkennung
- Election-Algorithmus auf (unidirektionalem) Ring
  - Bully-Algorithmus: nur grösste Identität schafft Ringumlauf → ist "gewählt"
  - Chang/Roberts-Algorithmus: message extinction (beim Ring kein Terminierungserkennungsproblem!)
- Chang/Roberts-Algorithmus auf unidirektionalem Ring
  - Worst-Case-Nachrichtenkomplexität: $O(n^2)$
- Chang/Roberts-Algo.: Mittlere Nachrichtenkomplexität?
  - Vermutung: i-t grösster macht im Mittel n/i Schritte
  - mittlere Nachrichtenkomplexität wäre dann nH_n (= ca. n ln n)
  - Wahrscheinlichkeit, genau i Positionen weit zu kommen
- Bidirektionale Varianten des Chang/Roberts-Algorithmus
  - probabilistisch
  - mittlere Nachrichtenkomplexität
- Algorithmus von Hirschberg und Sinclair (bidirekt. Ring)
  - sukzessive grössere Gebiete erobern
  - worst-case Nachrichtenkomplexität $< 8\ n \log_2 n$

Resümee (9)

- Petersons Election-Algorithmus (bidirektionaler Ring)
  - solange sukzessive Identität in beide Richtungen senden, bis man von einem grösseren Nachbarn erfährt
  - worst-case Nachrichtenkomplexität ca. $2 n \log_2 n$
  - mittlere Nachrichtenkomplexität ca. $2 n \log_3 n$
  - Simulation ("kostenneutral"!) auf einem unidirektionalen Ring
  - Variante: Ringrichtung jede Phase alternieren (spart insgesamt Nachrichten)

- Election auf Bäumen
  - Vereinigte Explosionswelle wird an den Blättern reflektiert
  - Kontraktionsphase propagiert Maximum; endet in zwei Zentrumsknoten
  - Nachrichtenkomplexität O(n)

- Echo-Election auf allgemeinen Graphen
  - Idee wie Chang/Roberts (d.h. message exctinction), aber Echo-Algorithmus statt Ringumlauf

- Nachrichtenkomplexität des Election-Problems
  - mindestens e Nachrichten

- Verteilte Spannbaumkonstruktion
  - Zusammenhang zum Election-Problem ("gleich schwierig")

- Anonyme Netze
  - De-Anonymisierung (bei eindeutigem Initiator)

- Election in anonymen Netzen
  - kein stets terminierender (deterministischer) Algorithmus möglich

Resümee (10a)

- Election in anonymen Netzen
  - kein stets terminierender (deterministischer) Algorithmus möglich
- Probabilistische Algorithmen
  - Las Vegas (terminiert nicht immer, Ergebnis ist aber korrekt)
  - Monte Carlo (terminiert, aber evtl. mit falschem Ergebnis)
- Probabilistische Election-Algorithmen
  - Las-Vegas-Verfahren mit Zufallsidentität

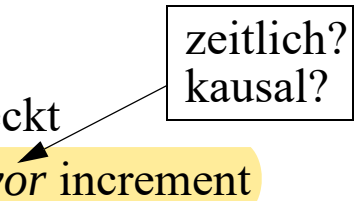
- Garbage-Collection: Modellierung
  - Objekte und Zeiger; Wurzelobjekte
  - nicht mehr von der Wurzel erreichbar → Garbage
  - rekursives Freigeben (Zyklen bleiben evtl. übrig!)
  - *Mutator* (new, copy, delete: Manipulation von Zeigern)
  - *Collector* soll Garbage-Objekte identifizieren
- Garbage-Collection: Grundverfahren
  - Paradigmen: "stop the world" / on the fly (= "parallel")
  - "Mark and sweep"-Verfahren
  - bei paralleler Variante: Problem mit "behind the back copy"
    ⇒ Mutator / Collector müssen sich koordinieren!
    (sonst bekäme der Collector u.U. ein "schiefes Bild")

Resümee (10b)

- Verteiltes Garbage-Collection (= GC in verteilten Systemen)
  - Unterschied zwischen lokalen und "remote" Referenzen
  - Referenzen u.U. "in transit"
  - copy nicht mehr atomar ("send/receive copy")
  - increment / decrement per Nachricht (z.B. an den Ort des Referenzzählers)
  - inc bzw. dec daher nicht "gleichzeitig" mit copy bzw. delete
  - lokales und globales GC (dezentral, echt parallel, typw. hierarchisch)

- Formalisierung des GC-Problems: Operationen C_p, R_p, D_p

- Referenzzähler-Verfahren
  - Problem: "zyklischer Garbage" wird nicht entdeckt
  - bei verteilter Variante: Problem bei decrement *vor* increment



Resümee (10c)

- Lösungen für verteiltes Reference-Counting:
  - prinzipiell: Causal Order garantieren (d.h. indirekte Überholungen vermeiden)
  - "naiv": auf Bestätigung jeder Increment-Nachricht warten
  - Varianten von Lermen/Maurer und Rudalics (zwei bis vier Nachrichten pro copy-Operation)

- Weighted Reference Counting (WRC)
  - Kopieren ohne Zusatznachricht: Splitten des Referenzgewichts

- Analogie (verteiltes) GC $\Leftrightarrow$ verteilte Terminierung

- Transformation GC-Algorithmus $\rightarrow$ Algorithmus zur Erkennung der verteilten Terminierung
  - Umformung des Terminierungsproblems in ein GC-Problem
  - darauf gegebenen GC-Algorithmus ansetzen

Resümee (11)

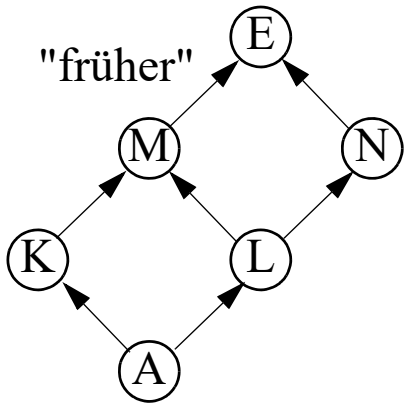
- *Garbage Collection*: Local Reference Counting (LRC)

- jede Maschine besitzt für *jedes* Objekt einen (lokalen) Zähler
- logische Baumstruktur ("Verantwortlichkeit")
- Nachrichten nur für maschinenübergreifendes copy oder decrement, wenn lokal keine Refernz auf entferntes Objekt mehr vorhanden
- IRT / ORT-Tabellen: gesplittete"Proxy-Objekte"; Referenzbündelung
- Migration von Objekten "leicht" zu unterstützen

- *Verteilte Berechnungen*: Formale Definition (Modellierung!)

- Partition von Ereignissen, Sende/Empfangsereignisse, Kausalrelation
- Zeitdiagramme von verteilten Berechnungen; Gummibandtransformation
- Globale Zustände als Endzustände von Präfixberechnungen (Präfixberechnungen sind linksabgeschlossen bzgl. der Kausalrelation)
- Menge der Zustände (bzw. Präfixberechnungen) bilden Verband

Berechnung läuft entlang eines "unbestimmten" Weges vom Anfangszustand A zum Endzustand E.



Resümee (12)

- Wellenalgorithmen
  - Information verteilen / einsammeln; Phasen trennen; Ereignisse triggern...
  - Formale Def: ... init < $visit_i$ < conclude ...
  - Bsp.: Echo-Algorithmus, Ring, Stern...
  - min. n-1 Nachrichten; min e Nachrichten bei unbekannten Nachbarn
  - Spannbaum = jeweils erste empfangene Nachricht eines Knotens
  - Visit-Ereignisse bilden einen *Schnitt* (wann senkrechte Schnittlinie möglich?)

- Virtuell gleichzeitiges Markieren mittels flooding
  - Voraussetzung: FIFO-Kanäle
  - "konsistente" Schnittlinien lassen sich senkrecht zeichnen
  - keine Nachricht läuft "rückwärts" über die Schnittlinie

- Sequentielle Traversierungsverfahren
  - spezielle Wellenalgorithmen: visit-Ereignisse linear geordnet

- Algorithmus von Tarry (Labyrinth-Problem)
  - Beweisidee, dass Tarry-Algorithmus ein Traversierungsverfahren ist
  - Depth-first-Search ist Spezialfall des Tarry-Algorithmus

- Traversierungsverfahren von Awerbuch
  - Variante von Cidon

Resümee (13)

- Globale konsistente Schnitte / Zustände
  - "konsistente" Schnittlinien lassen sich senkrecht zeichnen
  - keine Nachricht läuft "rückwärts" über die Schnittlinie

- Schnappschussalgorithmen

  (1) Färben von Prozessen / Nachrichten; Vermeiden von "Tachyonen"; In-transit-Nachrichten durch Weiterleiten von Kopien an den Initiator

  (2) Chandy/Lamport-Algorithmus: Flooding; FIFO-Kanäle ("flushing"); Problem (?): einige Kanäle sind scheinbar immer leer

- Beobachten verteilter Berechnungen
  - Wunsch: lückenlos konsistente Schnappschüsse anzeigen
  - rekonstruiertes Bild des Beobachters
  - ideale und kausaltreue Beobachter

- Kausaltreues Beobachten
  - Beispiele für kausal inkonsistente Beobachtungen
  - Def. kausaltreuer Beobachter
  - Pfade im n-dimensionalen Zustandsgitter ("Hyperwürfel")

- Entdecken globaler Prädikate durch Beobachtung
  - Abhängigkeit von konkreten Beobachtungen ("possible worlds")
  - Wirkung von Handshake- und Barrier-Synchronisation

- Stabile Prädikate

Resümee (14)

- Lamport-Zeit

- Schnitte und Zeitstempel
  - Später- / Früher-Relation auf Schnitten
  - Definition konsistenter Schnitte als linksabgeschlossene Ereignismengen
  - Zeitstempel eines Ereignisses als Menge seiner kausalen Vorgänger (Repräsentation durch lokal letztes Ereignis → Vektorzeit)

- Vektorzeit
  - Interpretation: repräsentiert gesamte kausale Vergangenheit
  - Zeitstempelarithmetik
  - Implementierung (Supremum beim Empfang)
  - Isomorphie der Zeit- und Kausalstruktur

- Anwendung der Vektoruhren
  - kausaltreue Beobachtungen

- Relativistische Struktur der Vektorzeit