Lösungsvorschläge zur Theoretischen Übungsserie A

Verteilte Systeme im HS 07/08

Benedikt Ostermaier

Research Group for Distributed Systems
ETH Zürich

12. November 2007

R1 (4.2.2008): Korrektur Folie 21

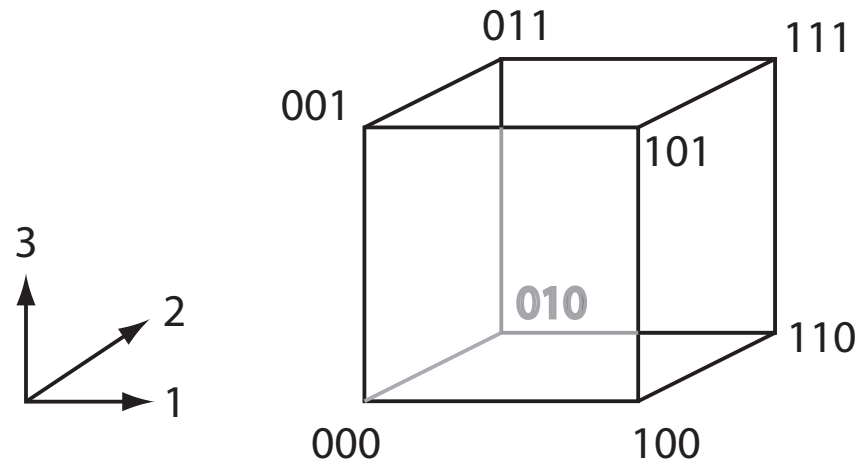
A1: Topologien und Pfadlängen (1)

A.1.1.) 256 Knoten, 16×16 Gitter

Maximale Pfadlänge: 30 Schritte

A.1.2) 256 Knoten, Hypercube

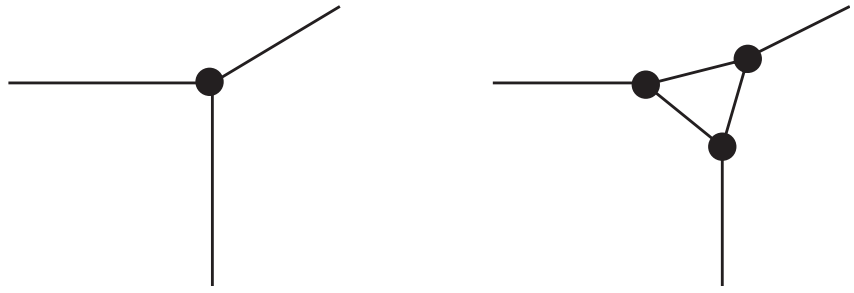
- Dimension des Hypercubes: $d = \log_2(256) = 8$



- Max. Anzahl von Schritten = Max. Anzahl Bitflips in der Adresse = $d = 8$

A1: Topologien und Pfadlängen (2)

Hypercube zu Cube Connected Cycle (CCC):



Ersetze jede Ecke durch einen Ring mit d Knoten

- Vorher: 2^d Knoten
- Nachher: $d \cdot 2^d$ Knoten

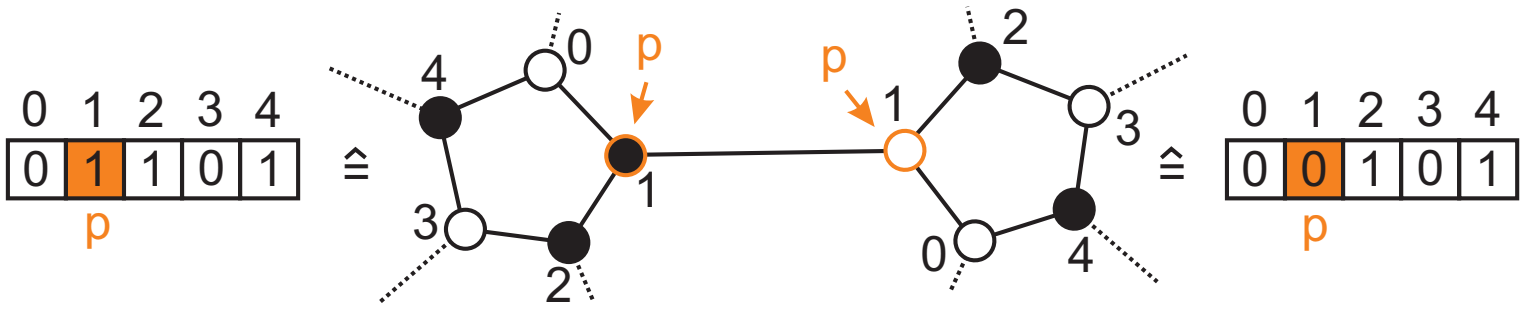
A.1.3.a) Anzahl Knoten eines CCC mit d=8

Hier also $8 \cdot 256 = 2048$ Knoten.

A1: Topologien und Pfadlängen (3)

A.1.3.b) Max. Pfadlänge im Cube Connected Cycle:

- Jeder Knoten hat eine CCC-Adresse, bestehend aus:
  - Hypercube-Adressteil: Nummer des Eckrings
  - Position p ($0 \leq p < d$) in seinem Eckring
- Wir betrachten einen "regulären" CCC, d.h. die Reihenfolge der Dimensionswechsel ist in jedem Eckring gleich und monoton!

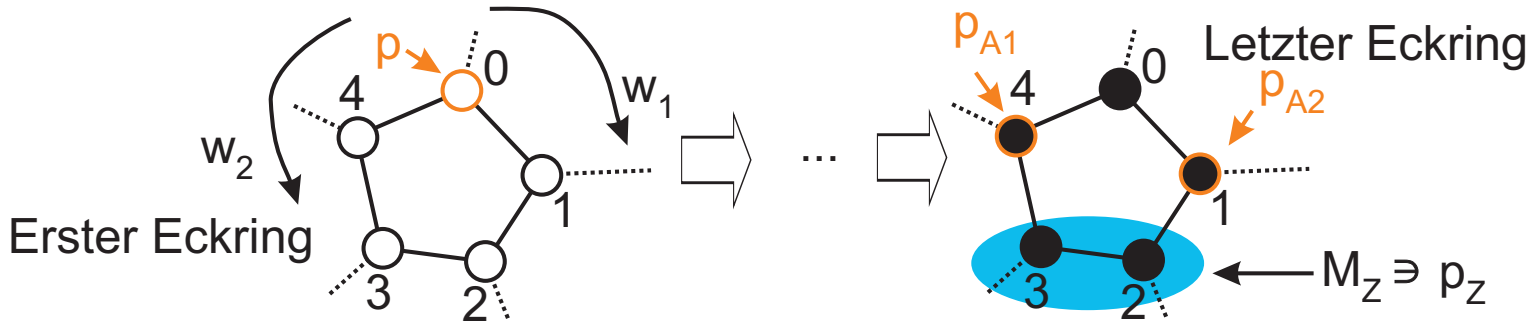


- Pro Schritt 3 Möglichkeiten:
  - Dimensionwechsel der Dimension p
  - Zum linken Nachbarn $((p-1) \bmod d)$ im Eckring
  - Zum rechten Nachbarn $((p+1) \bmod d)$ im Eckring

A1: Topologien und Pfadlängen (4)

Grösstmöglicher Abstand zwischen zwei Knoten:

- Max. Unterschied beim Hypercube-Adressteil, d.h. d verschiedene Bits
- Im letzten Eckring: Die Position des Zielknotens p_Z ist max. entfernt von der Ankunftsposition p_{A1} bzw. p_{A2}.
  - p_{A1}: Bewegung in den Ringen im Uhrzeigersinn (w_1)
  - p_{A2}: Bewegung in den Ringen entgg. dem Uhrzeigersinn (w_2)



p_Z muss also in der Menge M_Z enthalten sein. Daher kann man zwei Kanten ignorieren. Die max. Weglänge im letzten Ring ist also $\lfloor \frac{d-2}{2} \rfloor = \lfloor \frac{d}{2} \rfloor - 1$ Schritte.

A1: Topologien und Pfadlängen (5)

Insgesamt:

- d Dimensionswechsel notwendig
- Je Dimensionswechsel ein Schritt im Ring (ausser dem ersten)!
- Im Zielring maximal $\lfloor \frac{d}{2} \rfloor - 1$ Schritte
- Gesamt: $d + d - 1 + \lfloor \frac{d}{2} \rfloor - 1 = 2d + \lfloor \frac{d}{2} \rfloor - 2$
- D.h. für den Fall $d = 8$:
  $2 * 8 + \lfloor \frac{8}{2} \rfloor - 2 = 18$

A2: Pfade im Hypercube (1)

A.2.1) Wieviele Pfade gibt es zwischen zwei Knoten?

- Annahme: k Bits unterscheiden sich in Quell- und Zieladresse
- Im ersten Schritt k Möglichkeiten, im zweiten $(k-1)$, ...
- Das ergibt $k!$ Pfade

A2: Pfade im Hypercube (2)

Gegeben: Zwei Knoten im Abstand k.

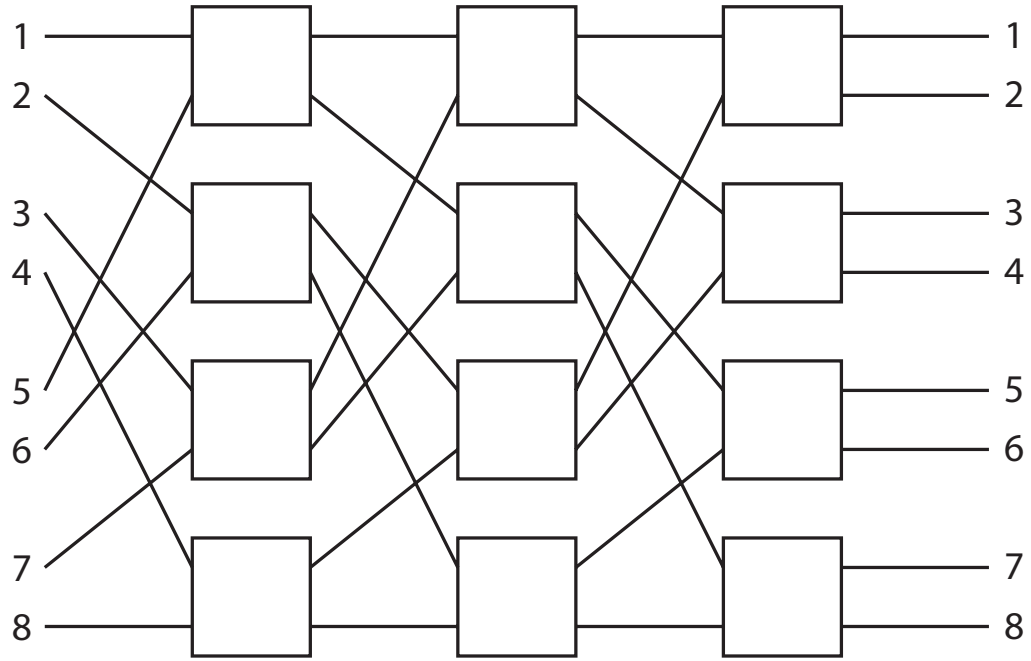
Anzahl der *knotendisjunkten* Pfade?

Behauptung: k solche Pfade

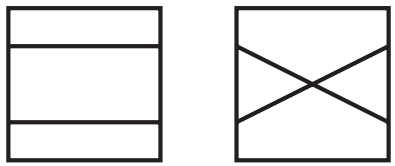
- Maximal k knotendisjunkte Pfade, denn Startknoten kann auf dem Weg zum Ziel zwischen k Nachbarn wählen
- Mindestens k Pfade *existieren:*
  - Pfad ist gegeben durch Folge von Dimensionswechseln (z.B. 1,4,3,2)
  - Anordnung der Dimensions*wechsel*, so dass sich keine gemeinsamen Zwischenknoten auf verschiedenen Pfaden ergeben
  - Start mit $n_1, n_2, \ldots, n_k$
  - 1. Rotieren um eine Stelle nach links: $n_2, n_3, \ldots, n_k, n_1$
  - 2. Rotieren: $n_3, n_4, \ldots, n_k, n_1, n_2$
  - $\ldots$
  - $(k-1)$. Rotieren: $n_k, n_1, \ldots, n_{k-1}$
  - Präfixe dieser Pfade enthalten alle unterschiedliche Elemente!

A3: Permutationsnetze (1)

Diese Netztopologie nicht üblich für verteilte Systeme, sondern für Bussysteme

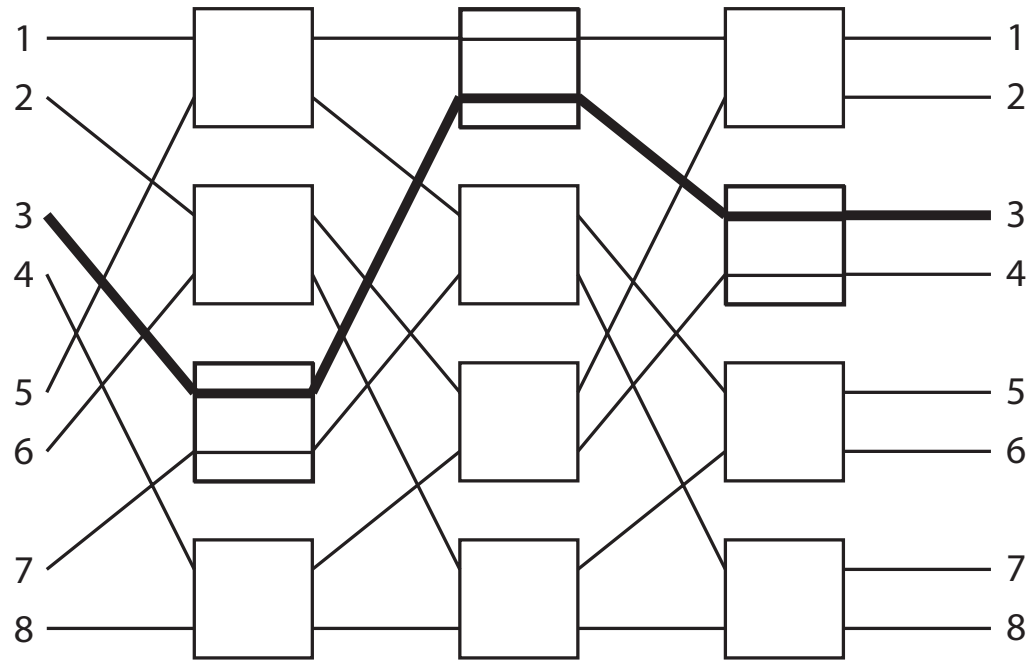


Mit diesen Schaltelementen kann jede gewünschte Verbindung hergestellt werden:



A3: Permutationsnetze (2)

Beispiel für die Verbindung von Eingang 3 nach Ausgang 3:



A3: Permutationsnetze (3)

A3.1.) Gegeben: n Eingänge (und ebensoviele Ausgänge).

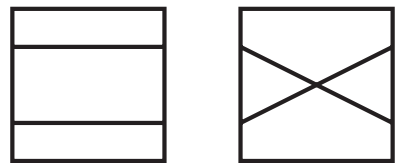
Warum sind $\log n$ Schaltstufen notwendig?

- Pro Stufe verdoppelt sich die Anzahl der Nachfolger
- Nach m Stufen hat man 2^m erreichbare Positionen
- Daher $\log_2 n$ Stufen notwendig, um n Positionen zu erreichen

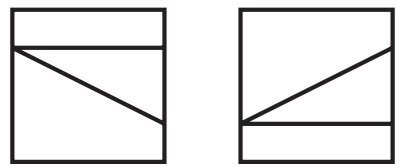
Warum sind jeweils $\frac{n}{2}$ Schalter notwendig?
Je Schalter 2 Ein- bzw. Ausgänge. Auf jeder Stufe müssen alle n Positionen verschaltet werden, daher $\frac{n}{2}$ Schalter je Stufe.

A3: Permutationsnetze (4)

A3.2.) Broadcast mit Permutationsnetzen

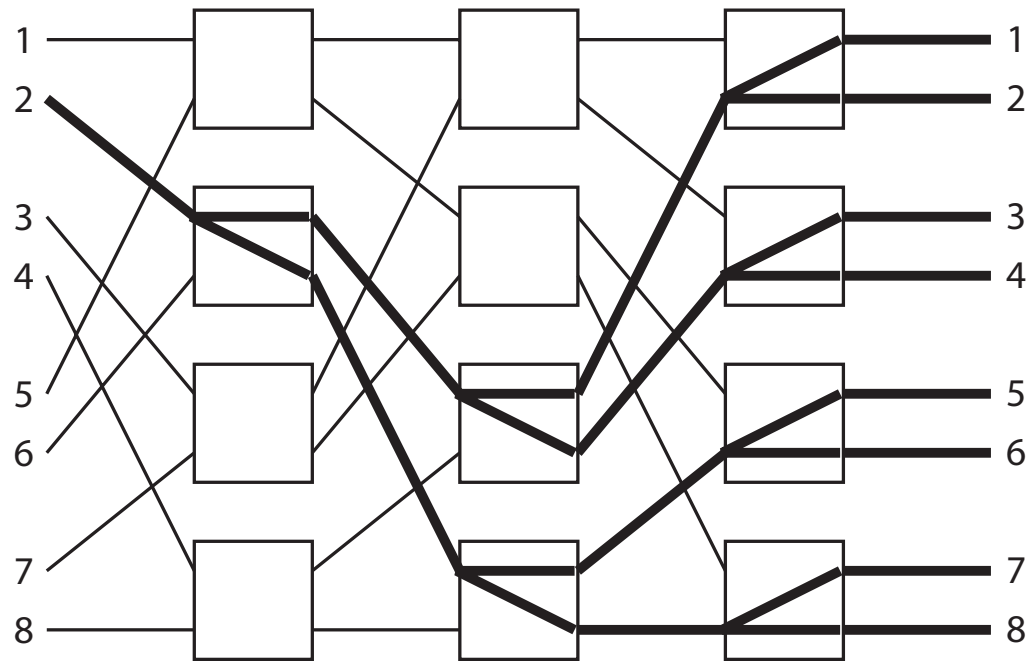


Broadcast mit Standard-Schaltelementen nicht möglich, denn pro Eingang nur ein Ausgang erreichbar.



Diese Elemente erlauben, einen Eingang mit beiden Ausgängen zu verknüpfen. Damit ist Broadcast möglich!

A3: Permutationsnetze (5)



A4: Fehlermodelle (1)

Anmerkung: Nicht alle Fehlerfälle lassen sich eindeutig einer bestimmten Fehlerklasse zuordnen.

1. (Webserver) Fail-Stop. Sender: Browser, Empfänger: Server. Der Server antwortet mit einer Fehlermeldung.
2. (Mobiltelefon) Crash bzw. Fail-Stop. Sender: GSM-Netz/anderer Teilnehmer, Empfänger: Mobiltelefon. Das Netz erkennt "Verlust" durch Timeout.
3. (Virus) Byzantinischer Fehler. Das Verhalten des Virus ist beliebig. Dass ein Fehler auftritt, kann nicht in jedem Fall festgestellt werden.
4. (W-LAN) Fehlerhaftes Senden und/oder Empfangen bzw. fehlerhaftes Übertragen. Hier ist der Link selbst betroffen, nicht einer der Kommunikationsteilnehmer.

A4: Fehlermodelle (2)

5. (E-Mail) Zeitfehler. Der Empfänger (Benutzer) erhält die Nachricht zu spät.
6. (Spam-Filter) Fehlerhaftes Empfangen. Die Nachricht (Mail) wird vom Empfänger (Spam-Filter) falsch behandelt. Der Sender merkt davon nichts.
7. (Drucker) Byzantinischer Fehler oder fehlerhaftes Empfangen. Entweder hat der Druckertreiber einen Bug (byz. Fehler) oder bei der Übertragung zum Drucker gehen Daten verloren, so dass der Drucker die Postscript-Datei nicht als solche erkennt.

A5: Fehlertoleranz

Gegeben: Zuverlässigkeit eines Dienstes pro Rechner bei 60%.

A5.1.) Zuverlässigkeit bei 3 Rechnern?

$1 - (1 - 0.6)^3 = 0.936 = 93.6\%$

A5.2.) Benötigte Replikate um 99% Verfügbarkeit zu erreichen?

$1 - (1 - 0.6)^n = 0.99 \Rightarrow 0.01 = 0.4^n \Rightarrow n = \log_{0.4}(0.01) = 5.03$
Es werden also $\lceil 5.03 \rceil = 6$ Rechner (5 Replikate) benötigt.

A5.3.) Ausfallzeit in Stunden/Jahr bei 99% Verfügbarkeit?

$(1 - 0.99) * (365 * 24) = 87.6$ Stunden

A6: Synchrone Kommunikation mit Mars-Rover?

Wie schnell darf der Mars-Rover maximal fahren?

- Signallaufzeit Erde↔Mars:
  55.7 Mio km / 300'000 km/s $\approx$ 186s.
- Benötigte Zeit um auf ein Hindernis zu reagieren:
  $2 *$ Signallaufzeit $+ 4s$ Reaktionszeit $=$ 376 Sekunden.
- Max. erkennbare Distanz zum Hindernis: 10m

$\Rightarrow$ In 376 Sekunden darf der Rover max. 10m zurücklegen, d.h.
$v_{max} = 10m/376s = 0.0266m/s = 96m/h$

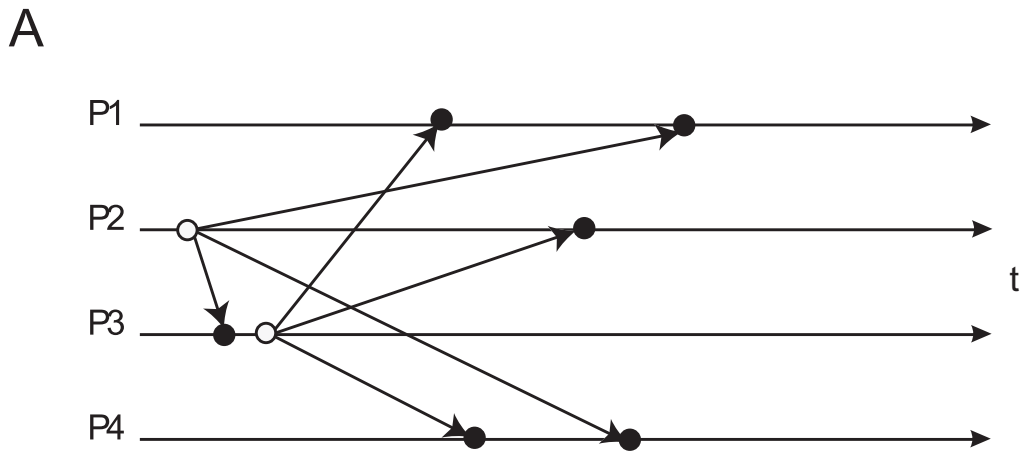
Bemerkung: ein echter Mars-Rover ist auf maximal 5 cm/s ausgelegt und bewegt sich real mit ca. 1 cm/s.

A7: RPC

1. Referenzen sind grundsätzlich nicht erlaubt, da eine Referenz (Pointer) auf Rechner A auf Rechner B keine Bedeutung hat. Das gilt sowohl für Eingabe- wie Ausgabeparameter. Man könnte jedoch die Datenstruktur hinter der Referenz sequentialisieren und in dieser Form übertragen.
2. Es besteht die Gefahr, dass eine Anfrage mehrfach bearbeitet wird. Sequenznummern helfen.
3. At-least-once

A8: Broadcast (1)

A.8.1) Atomarer Broadcast? Beispiel A



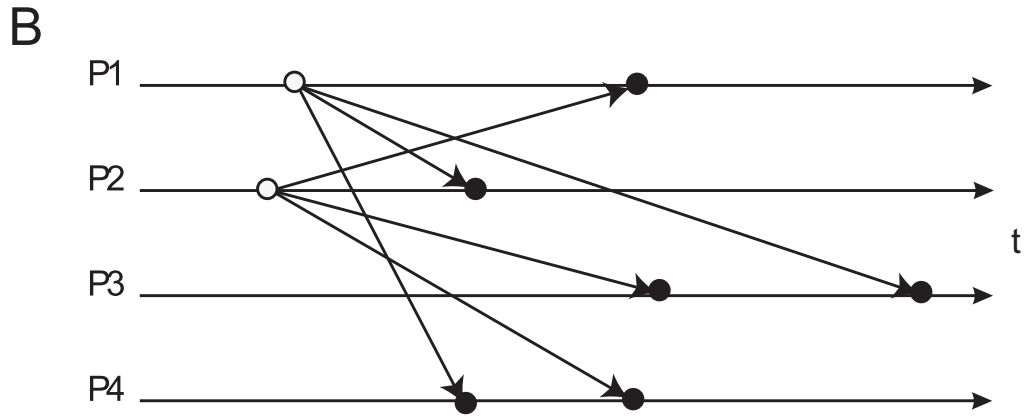
Def. atomarer (totaler) Broadcast: wenn P_1 und P_2 die Nachrichten N, M erhalten, ist die Empfangsreihenfolge bei beiden Prozessen die gleiche.

Beachte: Das Senden einer Nachricht zählt nicht als Empfang!

Hier: Empfangsreihenfolge bei P_1 und P_4 gleich $\Rightarrow$ **atomarer Broadcast**

A8: Broadcast (2)

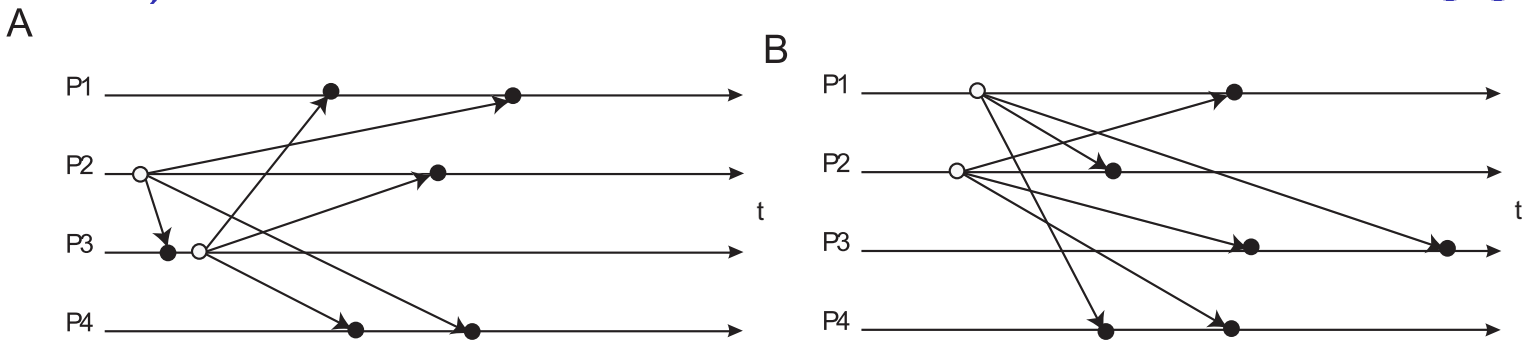
A.8.1) Atomarer Broadcast? Beispiel B



Empfangsreihenfolge bei P_3 und P_4 unterschiedlich $\Rightarrow$ **kein atomarer Broadcast**

A8: Broadcast (3)

A.8.2) Sind die Broadcasts voneinander kausal abhängig?



Simpler Test auf kausale Abhängigkeit: gibt es einen Pfad (von links nach rechts), der vom Sendeereignis von X zum Sendeereignis von Y führt?

A: ja! **Aber:** kausale Reihenfolge ist nicht gewahrt, denn die Reihenfolge bei den Empfängern P_1 und P_4 entspricht nicht der Kausalität!

B: nein, keine kausale Abhängigkeit der Broadcasts

A8: Broadcast (4)

A.8.3) Totale Ordnung mit zentralem Sequencer

Frage: Sind Broadcasts, die über einen zentralen Sequencer gesendet werden, notwendigerweise total geordnet? Welche Bedingung muss gelten?

Falls Kanäle vom Sequencer zu den Empfängern **FIFO-Eigenschaft** besitzen, dann gilt totale Ordnung, denn: wenn Nachricht X vor Y gesendet, dann kommt X vor Y an (Nachrichten überholen sich auf FIFO-Kanal nicht)

FIFO-Eigenschaft kann z.B. über Acknowledgements hergestellt werden (Sequencer braucht aber ein ACK von *allen* Empfängern)

A9: Uhrensynchronisation I

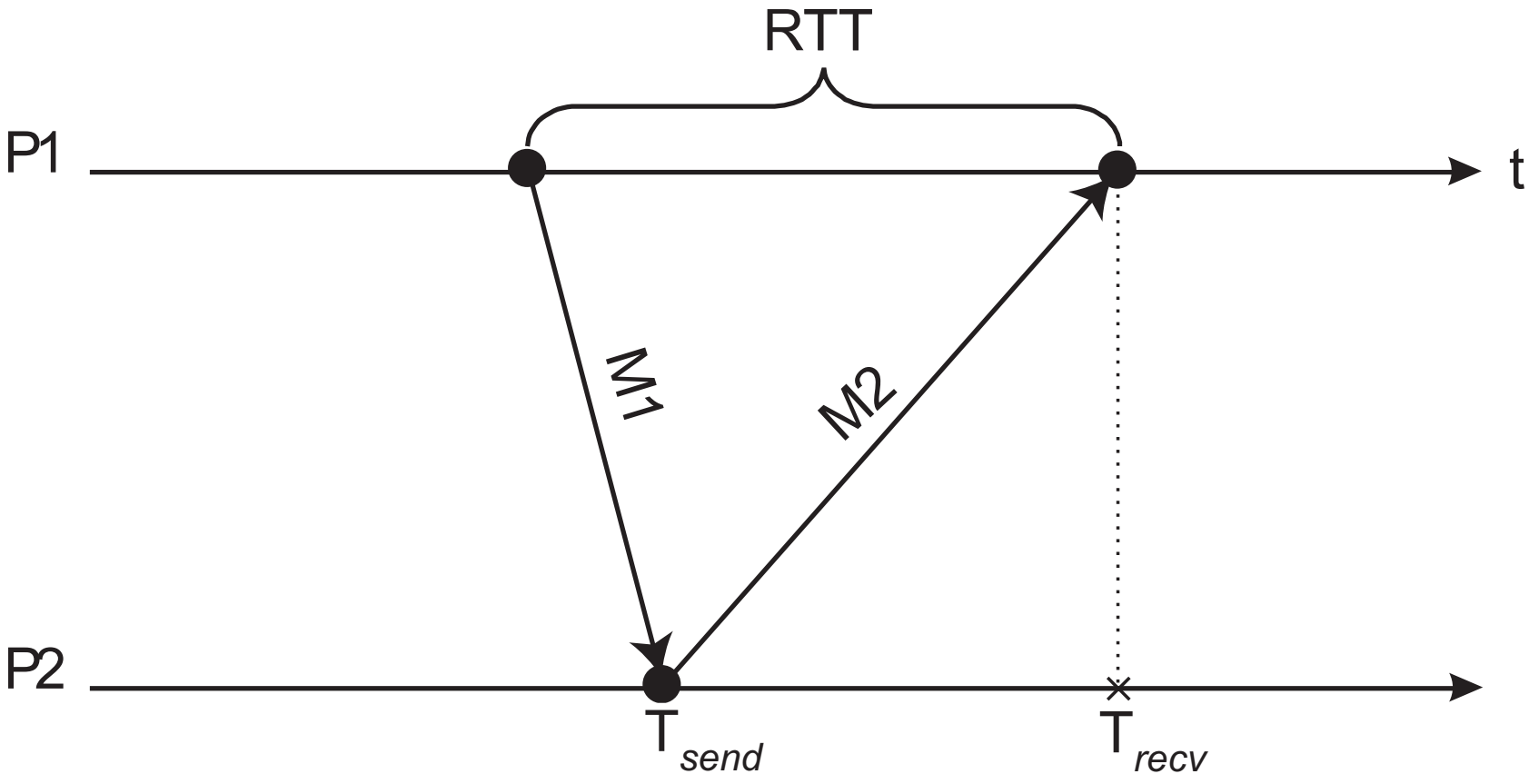
Situation

- Uhr tickt 1-mal je Millisekunde $\hat{=}$ 1000-mal pro Sekunde. Jede Minute erfolgt ein Update (Zurücksetzen der Uhr auf 0).
- Auf Rechner A arbeitet die Uhr korrekt. Auf Rechner B tickt die Uhr nur 990-mal pro Sekunde.

Update jede Minute; maximaler Unterschied nach 60s:

- `clock(A) = 60000, clock(B) = 59400 ticks`
- $\Delta ticks = 60000 - 59400 = 600$ ticks
- Maximaler Unterschied in der Zeitwahrnehmung: 600 ms.

A10: Uhrensynchronisation II (1)



A.10.1.) Gründe für Varianz der Nachrichtenlaufzeit

Schwankende Netzauslastung (Puffer), unterschiedliche Routen, ...

A10: Uhrensynchronisation II (2)

A.10.2.) Anpassen der Uhrzeit von P1 an die von P2 (1)

- Nachrichtenlaufzeit zwischen 10 und 35 ms.
- P_2 sendet auf Anfrage (M_1) Zeitstempel T_{send} in M_2
- Welche Uhrzeit sollte P_1 setzen? (Idealerweise T_{recv}, jedoch ist diese P1 nicht genau bekannt!)

Den Mittelwert, da hier die maximale Abweichung minimal ist.

- $T_{min} = T_{send} + 10ms$, $T_{max} = T_{send} + 35ms$
- $T_{mean} = \frac{T_{min}+T_{max}}{2} = T_{send} + 22.5ms$
- `clock(P1)` $:= T_{mean}$
- Maximale Abweichung von T_{mean}: $T_{mean} - T_{min} = 12.5ms$

A10: Uhrensynchronisation II (3)

A.10.3.) Anpassen der Uhrzeit von P1 an die von P2 (2)

- Nur minimale Laufzeit von Nachrichten (=10 ms) bekannt!
- P_2 sendet auf Anfrage (M_1) Zeitstempel T_{send} in M_2
- Welche Uhrzeit sollte P_1 setzen?

Möglicher Wertebereich für die Uhrzeit von P2:

- Untere Schranke (M_2 hat min. Laufzeit): $T_{min} = T_{send} + 10ms$
- Obere Schranke (M_2 hat max. Laufzeit):
  $T_{max} = T_{send} + RTT - 10ms$
- Mittelwert: $T_{mean} = \frac{T_{min}+T_{max}}{2} = T_{send} + \frac{RTT}{2}$
- Maximale Abweichung von T_{mean}:
  $T_{mean} - T_{min} = \frac{RTT}{2} - 10ms$

Stellen der Uhrzeit von P1 - wiederum auf den Mittelwert:

- `clock(P1)` $:= T_{mean} = T_{send} + \frac{RTT}{2}$

A10: Uhrensynchronisation II (4)

Beispiel:

- $RTT = 80ms$
- `clock(P1)` $:= T_{send} + 40ms$
- Untere Schranke für reale Zeit: $T_{send} + 10ms$ (Minimallaufzeit)
- Obere Schranke: $T_{send} + 70ms$ (Maximallaufzeit für M_2)
- Genauigkeit: $\pm 30ms$

Danke für Ihre Aufmerksamkeit

...jetzt folgt die Abnahme der ersten praktischen Aufgabe. Bitte begeben Sie sich dazu in die Räume C31 bzw. D31.