

Modul 63211 – Verteilte Systeme
 Einsendeaufgaben 3 – WS 2025/2026
 FernUniversität Hagen, Lehrgebiet Kooperative Systeme

Aufgabe 3.1: Sequentielle und Kausale Konsistenz 5 + 20 Punkte

Betrachten Sie die Ausführungsgeschichte der vier Prozesse P_1 , P_2 , P_3 und P_4 :

P_1 :	$W(x)1$	$W(x)3$	$W(y)1$		
P_2 :		$R(x)1$	$W(x)2$	$R(y)1$	$R(x)3$
P_3 :		$R(x)1$		$R(x)3$	$R(x)2$ $R(y)1$
P_4 :		$R(x)1$		$R(x)2$	$R(y)1$ $R(x)3$

1. Ist diese Ausführungsgeschichte *sequentiell konsistent*?
2. Ist diese Ausführungsgeschichte *kausal konsistent*?

Bitte begründen Sie Ihre Antworten.

Aufgabe 3.2: Replikation und Konsistenzmodelle 3 + 10 + 12 Punkte

In einem System von n Servern S_1, \dots, S_n mit replizierten Daten soll die *client-zentrierte Konsistenz* gewährleistet werden.

Clients schicken Anfragen mit Lese- oder Schreiboperationen an das System, die jeweils von genau einem der Server bearbeitet und beantwortet werden.

1. Welche Operationen müssen im System verbreitet werden?
2. Die Verbreitung der Operationen soll mit Hilfe von einem *Vektor-Uhr-System* realisiert werden.
 - Was bedeuten die einzelnen Komponenten der Vektor-Uhr?
 - Wie kann eine verbreitete Operation eindeutig im System identifiziert werden?
 - Wie gibt ein Server einem anderen Server Operationen weiter?
 - Wie aktualisiert ein Server seine Vektor-Uhr, wenn er Operationen von einem anderen Server erhalten hat?
3. Für alle vier Arten der client-zentrierten Konsistenz geben Sie jeweils bitte an, welche Informationen ein Client bei Anfragen mitliefern muss und was ein Server sicherstellen muss, bevor er die Antwort zurückschickt.

Aufgabe 3.3: Recovery 10 + 5 Punkte

Abbildung 1 zeigt die Ausführung der Prozesse P_0 , P_1 und P_2 , die durch Nachrichtenaustausch kommunizieren, dabei werden Nachrichten m_i , $i = 0, \dots, 7$ gesendet und empfangen. Prozess P_i setzt unabhängig die *Checkpoints* C_{ij} mit $i = 0, 1, 2$ und $j = 0, 1, 2$ ohne Koordination. Jeder Prozess startet mit einem initialen Checkpoint C_{i0} mit $i = 0, 1, 2$.

1. Gibt es einen *Dominoeffekt* (*domino effect*), wenn Prozess P_2 nach dem Senden von Nachricht m_6 ausfällt und eine *Wiederherstellung* (*rollback recovery*) startet?
2. Wo liegt die *Recovery line*?

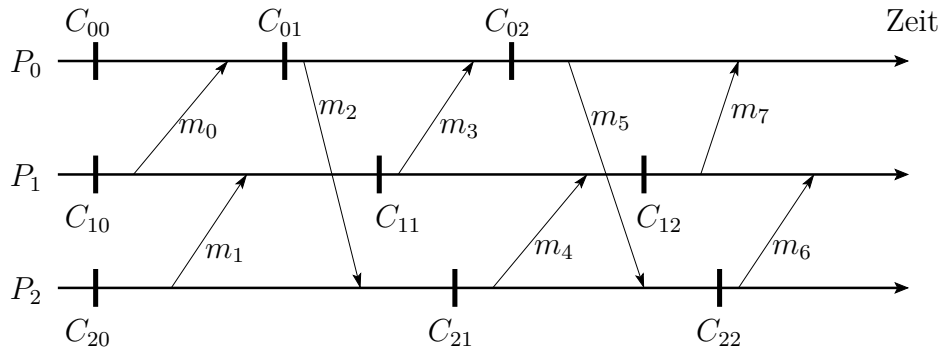


Abbildung 1: Interaktionen zwischen Prozessen P_0 , P_1 und P_2 .

Aufgabe 3.4: **Multicasting** **4 + 4 + 9 + 14 + 2 + 2 Punkte**

- In einer Gruppe von 3 Prozessen, P_1 , P_2 und P_3 , werden fünf *Multicast-Nachrichten* an die ganze Gruppe gesendet, wobei die Nachrichten 1 und 4 von P_1 kommen, Nachricht 2 von P_2 und die Nachrichten 3 und 5 von P_3 .
 - Wie viele Auslieferungsreihenfolgen der Nachrichten gibt es hier beim *atomaren Multicasting*?
 - Wenn wir jetzt voraussetzen, dass P_1 die Nachricht 1 vor 4 sendet und P_3 die Nachricht 3 vor 5, welche Auslieferungsreihenfolgen sind nun beim *FIFO-atomaren Multicasting* zulässig?
 - Außerdem nehmen wir noch an, dass P_1 die Nachricht 2 vor dem Absenden von 4 und P_3 die Nachricht 1 vor dem Absenden von 3 ausgeliefert hat. Welche Auslieferungsreihenfolgen der Nachrichten sind noch zulässig beim *kausalen atomaren Multicasting*? Bitte geben Sie diese Auslieferungsreihenfolgen an.
- Betrachten wir jetzt Abbildung 2, in der die Reihenfolge der Punkte auf einem Zeitstrahl die Reihenfolge des *Absendens* und *Empfangens* der Nachrichten darstellt.

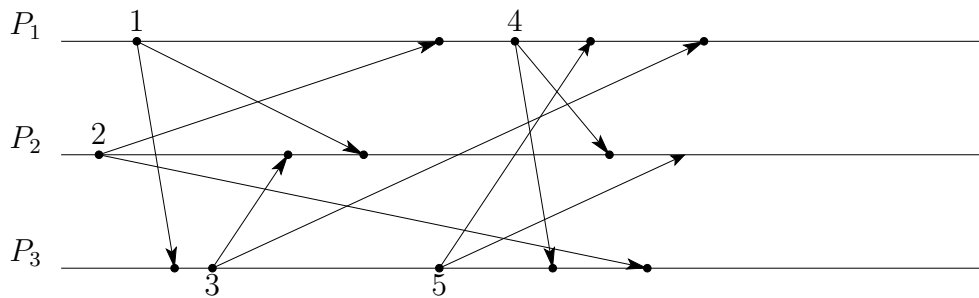


Abbildung 2: Kommunikation per Multicasting von drei Prozessen.

Prozesse P_1 , P_2 und P_3 führen den Algorithmus *zum Erzwingen kausaler Kommunikation* aus (siehe Note 6.4 ab Seite 321), um die Nachrichten der Anwendungsschicht in einer kausalen Ordnung *auszuliefern*.

- Geben Sie zu jeder Nachricht den Zeitstempel beim Senden und bei der Auslieferung an.
- In welcher Reihenfolge werden die fünf Nachrichten jeweils von P_1 , P_2 und P_3 ausgeliefert?
- Realisiert der Algorithmus ein *atomar kausales Multicasting*?