

## Aufgabenblatt 9

(Zu bearbeiten bis 08.01.)

## Vorlesung Verteilte Systeme Wintersemester 2025/26

### Aufgabe 1: Ring-Algorithmus

In der Vorlesung (Kap. 7.1) war bei den Wahl-Algorithmen ein Ring-Algorithmus dargestellt. Die Animation im Skript zeigt zwei ELECTION-Nachrichten, die gleichzeitig kreisen. Es ist zwar kein Problem, wenn es zwei solcher Nachrichten gibt, aber es wäre eleganter, wenn eine davon entfernt werden könnte.

Entwickeln Sie einen Algorithmus, der genau dies tut, ohne die Arbeitsweise des grundlegenden Wahl-Algorithmus zu stören.

### Aufgabe 2: Zentralisierter Algorithmus zum wechselseitigen Ausschluß

- a) Im zentralisierten Ansatz des wechselseitigen Ausschlusses (s. Vorlesung Kap. 7.2) erteilt der Koordinator nach Empfang einer Nachricht von einem Prozess, der seinen exklusiven Zugriff auf den kritischen Bereich freigibt, normalerweise dem ersten Prozess in der Warteschlange die Berechtigung. Nennen Sie einen anderen möglichen Algorithmus für den Koordinator.
- b) Angenommen, der Koordinator stürzt ab. Ist damit auch immer das gesamte System arbeitsunfähig? Falls nicht, unter welchen Umständen passiert dies? Gibt es eine Möglichkeit, das Problem zu vermeiden und das System in die Lage zu versetzen, Abstürze des Koordinators zu kompensieren?

### Aufgabe 3: Algorithmus von Ricart und Agrawala

Der Algorithmus von Ricart und Agrawala hat das Problem, dass falls ein Prozess abgestürzt ist und nicht auf eine Anforderung von einem anderen Prozess reagiert, in einem kritischen Bereich einzutreten, die fehlende Antwort als Verweigerung der Berechtigung interpretiert wird. Eine Methode, um abgestürzte Prozesse einfach zu erkennen, besteht darin, alle Anforderungen unmittelbar (ggf. negativ) zu beantworten. Gibt es Situationen, wo selbst diese Methode nicht ausreichend ist?

### Aufgabe 4: Deadlocks - Ricart und Agrawala

Ein verteiltes System kann mehrere voneinander unabhängige kritische Bereiche haben. Angenommen, Prozess 0 will in den kritischen Bereich A und Prozess 1 will in den kritischen Bereich B eintreten. Kann der Algorithmus von Ricart und Agrawala zu Deadlocks führen? Erklären Sie Ihre Antwort.

### **Pflichtaufgabe 5: Kausal sortierter Multicast** **Abgabe über moodle!**

Angenommen, ein System stellt zuverlässigen Multicast mit FIFO-Sortierung zur Verfügung. Man könnte auf die Idee kommen, darauf aufbauend einen kausal sortierten Multicast wie folgt zu implementieren:

- jede Nachricht wird mit dem Lamport-Zeitstempel des Sende-Ereignisses versehen,
  - die Empfänger-Knoten liefern ankommende Nachrichten in einer Reihenfolge an die Empfängerprozesse aus, die konsistent mit der Ordnung der Zeitstempel ist.
- a) Begründen Sie, warum dieser Multicast kausal sortiert ist.
- b) Wo ist das Problem bei dieser Implementierung? (**Tip**: wann kann eine Nachricht an den Empfängerprozess ausgeliefert werden?) Wie könnte es gelöst werden?
- c) Begründen Sie, warum dieser Multicast nicht vollständig sortiert ist. Wie könnte man eine vollständige Sortierung erreichen?

## **Pflichtaufgabe 6: Programmierung: Remote Write Protokoll**

### **Abgabe über moodle!**

In dieser Aufgabe sollen Sie mit Hilfe des Verteilte-Systeme-Simulators das Remote Write Protokoll für sequentielle Konsistenz (siehe Abschnitt 9.3.2 der Vorlesungsfolien) implementieren.

Bearbeiten Sie dazu die Aufgabe „[Primary Based Remote Write Protocol for Sequential Consistency](#)<sup>1</sup>“ im Wiki (Zugriff nur aus dem Uni-VPN!).

---

<sup>1</sup><https://git.bs.informatik.uni-siegen.de/dsbox/exercises/wiki/6-remote-write>