

Gegenseitiger Ausschluss

- Motivation
- Grundlagen
- Maekawa-Algorithmus



Grundsätzliche Herangehensweisen

- Nutzung eines zentralen Koordinators
 - Erteilung von Eintrittserlaubnissen durch einen separaten Dienst
 - Bei Bedarf: Replikation des zentralen Koordinators
- Tokenbasierte Algorithmen
 - Eintrittserlaubnis wird durch eine Marke (*Token*) repräsentiert
 - Problem: Wie kommt ein eintrittswilliger Knoten in den Besitz der Marke?
- Vollständig verteilte Algorithmen
 - Einholen der Erlaubnisse aller Knoten des verteilten Systems
 - Beispiel: Lock-Protokoll von Lamport
- [Siehe 6. Übungsaufgabe.]
- Quorenbasierte Algorithmen
 - Dezentraler Ansatz mit reduziertem Kommunikationsaufwand
 - Einholen der Erlaubnisse einer Untermenge von Knoten
 - Beispiel: Maekawa-Algorithmus



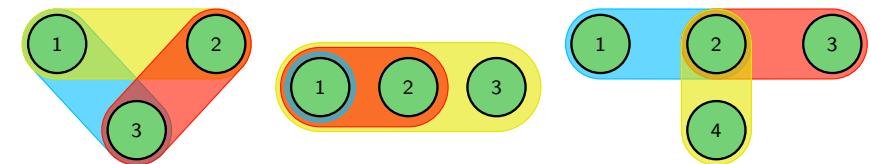
Motivation

- Problemstellung
 - Zugriff auf gemeinsame Ressourcen durch mehrere Knoten
 - Koordinierung des Eintritts in einen *kritischen Abschnitt*
 - Zu keinem Zeitpunkt darf mehr als ein Knoten die Eintrittserlaubnis haben
- Zusätzliche Anforderungen in der Praxis (Beispiele)
 - Fairness bei der Erteilung der Eintrittserlaubnis
 - Tolerierung von Knotenfehlern
 - Effiziente Nutzung des Netzwerks
 - Geringe Koordinierungsverzögerung zwischen
 - Zeitpunkt der Freigabe des kritischen Abschnitts durch einen Knoten und
 - Zeitpunkt des Betretens des Abschnitts durch einen anderen Knoten
- Herausforderungen
 - Wie lässt sich gegenseitiger Ausschluss in einem verteilten System erzielen?
 - Wie kann die erforderliche Anzahl an Nachrichten klein gehalten werden?



Quorensysteme

- Charakteristika eines Quorensystems
 - Menge von Knotenmengen („*Quoren*“)
 - Quoren überschneiden sich paarweise in mindestens einem Element
- Varianten und Spezialfälle
 - Zentraler Koordinator
 - Einzelnes Quorum mit allen Knoten
 - Alle Untermengen mit mehr als der Hälfte aller Knoten (*Mehrheitsquoren*)
 - Gewichtete Mehrheitsquoren
 - Jedem Knoten i wird ein Gewicht G_i zugewiesen
 - Gesamtgewicht aller Knoten $G_{total} := \sum_i G_i$
 - Quoren: Alle Knotenmengen mit Gewicht $G_{quorum} > \frac{G_{total}}{2}$



Grundansatz

- Dezentraler Algorithmus zum gegenseitigen Ausschluss von N Knoten
- Minimierung des Kommunikationsaufwands durch Nutzung von Quoren
- Einsatz total geordneter Zeitstempel aus Knoten-ID und Sequenznummer
- Automatisierte Erkennung und Auflösung von Verklemmungen

Nachrichtenaufwand pro kritischem Abschnitt bei Quorengröße $|S|$

- $3(|S| - 1)$ Normalfall
- $4(|S| - 1)$ Günstiger Konfliktfall
- $5(|S| - 1)$ Ungünstigster Fall

$\rightarrow O(\sqrt{N})$ Nachrichten

Literatur



Mamoru Maekawa

A \sqrt{N} algorithm for mutual exclusion in decentralized systems
ACM Transactions on Computer Systems, 3(2):145–159, 1985.



Anforderung und Freigabe des kritischen Abschnitts

- Versand einer Anforderung $\langle \text{REQUEST}, i, t \rangle$ durch Knoten i
 - $t := (i, s)$: Tupel aus Knoten-ID und lokaler Sequenznummer
 - Empfänger: Alle Knoten des Anforderungsquorums S_i
- Empfang einer Anforderung $\langle \text{REQUEST}, i, t \rangle$ durch Knoten j
 - R1. Hinzufügen der REQUEST zur Anforderungsmenge
 - R2. Überprüfung der Belegungskennung
 - „Frei“: Wechsel zu „Belegt von i_t “ und Senden von $\langle \text{LOCKED}, j, t \rangle$ an i
 - „Belegt von k_t “: Verzögertes Senden der Erlaubnis [Siehe Folie 10–9.]
- Empfang einer Erlaubnis $\langle \text{LOCKED}, j, t \rangle$ durch Knoten i
 - L1. Hinzufügen von Knoten j zur Erlaubnismenge E_i
 - L2. Betreten des kritischen Abschnitts, falls $|E_i| = |S_i|$
- Freigabe: $E_i := \emptyset$ und Senden von $\langle \text{RELEASE}, i, t \rangle$ an alle Knoten in S_i
- Empfang einer Freigabe $\langle \text{RELEASE}, i, t \rangle$ durch Knoten j
 - S1. Entfernen der zugehörigen Anforderung aus der Anforderungsmenge A_j
 - S2. Bearbeitung der ältesten Anforderung aus A_j [Siehe Schritt R2.]



Nachrichten

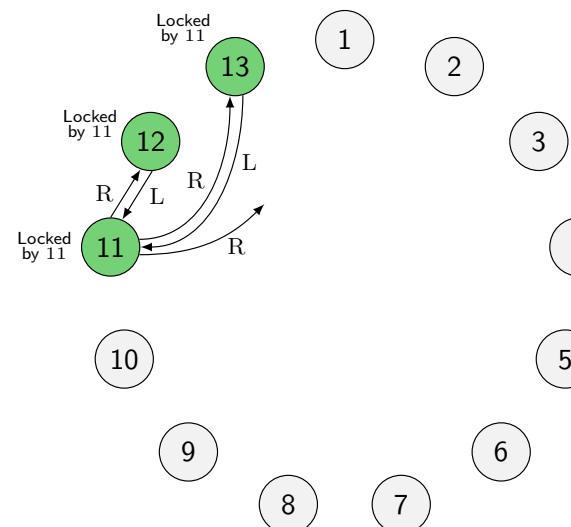
| Nachrichtentyp | Beschreibung |
|----------------|--|
| REQUEST | Anforderung zum Eintritt in den kritischen Abschnitt |
| LOCKED | Eintrittserlaubnis für den kritischen Abschnitt |
| RELEASE | Freigabe des kritischen Abschnitts |
| FAILED | Ablehnung einer Anforderung |
| INQUIRE | Bitte um die Rückgabe einer Erlaubnis |
| RELINQUISH | Rückgabe einer Erlaubnis |

Datenstrukturen eines Knotens i

- Anforderungsmenge Anforderungen, die i bekannt sind
- Anforderungsquorum Knoten, die i selbst um Erlaubnis fragen muss
- Erlaubnismenge Knoten, die i eine Erlaubnis gesendet haben
- Belegungskennung
 - „Belegt von j “
 - „Frei“



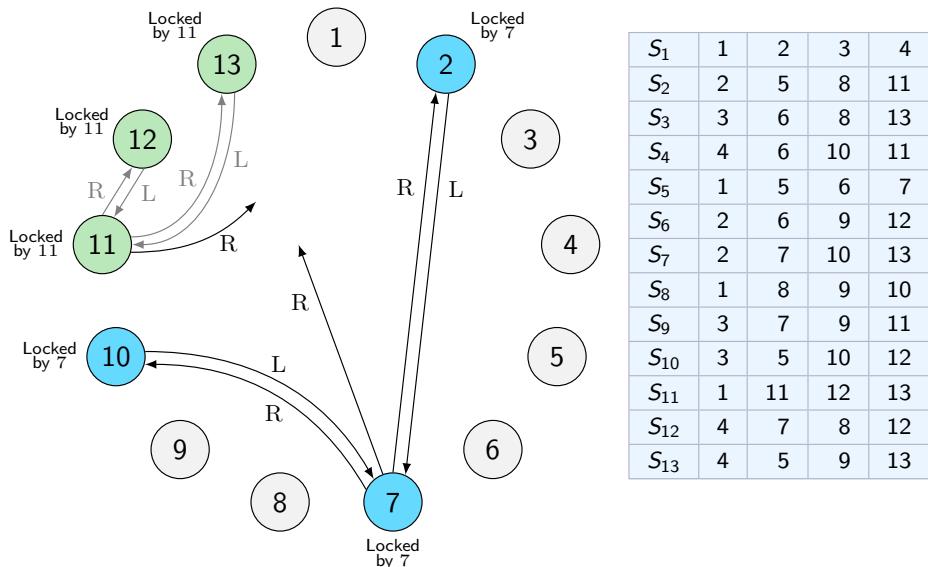
Beispiel



| | | | | |
|----------|---|----|----|----|
| S_1 | 1 | 2 | 3 | 4 |
| S_2 | 2 | 5 | 8 | 11 |
| S_3 | 3 | 6 | 8 | 13 |
| S_4 | 4 | 6 | 10 | 11 |
| S_5 | 1 | 5 | 6 | 7 |
| S_6 | 2 | 6 | 9 | 12 |
| S_7 | 2 | 7 | 10 | 13 |
| S_8 | 1 | 8 | 9 | 10 |
| S_9 | 3 | 7 | 9 | 11 |
| S_{10} | 3 | 5 | 10 | 12 |
| S_{11} | 1 | 11 | 12 | 13 |
| S_{12} | 4 | 7 | 8 | 12 |
| S_{13} | 4 | 5 | 9 | 13 |



Beispiel



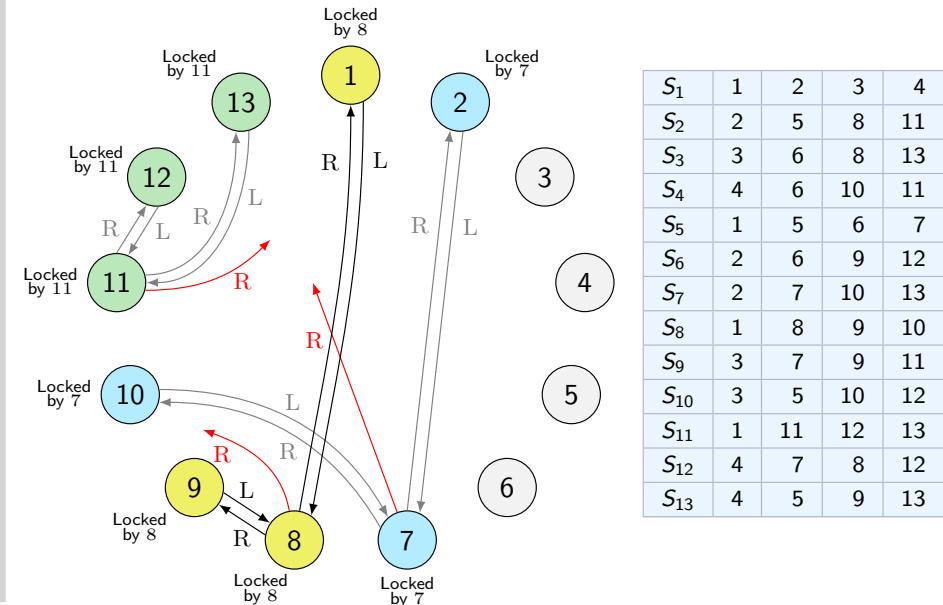
© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10-8

Erkennung und Auflösung von Verklemmungen

- Empfang einer Anforderung $\langle \text{REQUEST}, i, t \rangle$ durch Knoten j
 - Annahme: Belegungskennung von Knoten j ist „Belegt von $k_{t'}$ “
 - Vorgehensweise
 - Senden von $\langle \text{INQUIRE}, j, t' \rangle$ an k falls $\langle \text{REQUEST}, i, t \rangle$ älteste Anforderung
 - Senden von $\langle \text{FAILED}, j, t \rangle$ an i sonst
 - Empfang von $\langle \text{INQUIRE}, j, t' \rangle$ durch Knoten k
 - Falls k zuvor ein $\langle \text{FAILED}, x, t \rangle$ von einem Knoten x empfangen hat
 - Ia1. Entfernen von Knoten j aus der Erlaubnismenge
 - Ia2. Senden von $\langle \text{RELINQUISH}, k, t' \rangle$ an Knoten j
 - Sonst
 - Ib1. Speicherung der INQUIRE-Nachricht
 - Ib2. Ausführung der Schritte Ia1 und Ia2 bei Empfang von $\langle \text{FAILED}, x, t \rangle$

Beispie



© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10-8

Erkennung und Auflösung von Verklemmungen

- Empfang einer Rückgabe $\langle \text{RELINQUISH}, k, t' \rangle$ durch Knoten j
 - Q1. Löschen der bisherigen Belegungskennung „Belegt von $k_{t'}$ “
 - Q2. Aktualisierung der Belegungskennung
 - Annahme: $\langle \text{REQUEST}, i, t \rangle$ ist ältestes Element in der Anforderungsmenge
 - Neue Belegungskennung: „Belegt von i_t “
 - Q3. Senden der Erlaubnis $\langle \text{LOCKED}, j, t \rangle$ an Knoten i
 - Empfang einer Ablehnung $\langle \text{FAILED}, j, t \rangle$ durch Knoten i
 - Falls i zuvor ein $\langle \text{INQUIRE}, x, t \rangle$ empfangen hat
 - Fa1. Entfernen von Knoten x aus der Erlaubnismenge
 - Fa2. Senden von $\langle \text{RELINQUISH}, i, t' \rangle$ an Knoten x
 - Sonst
 - Fb1. Speicherung der FAILED-Nachricht
 - Fb2. Ausführung der Schritte Fa1 und Fa2 bei Empfang von $\langle \text{INQUIRE}, x, t \rangle$

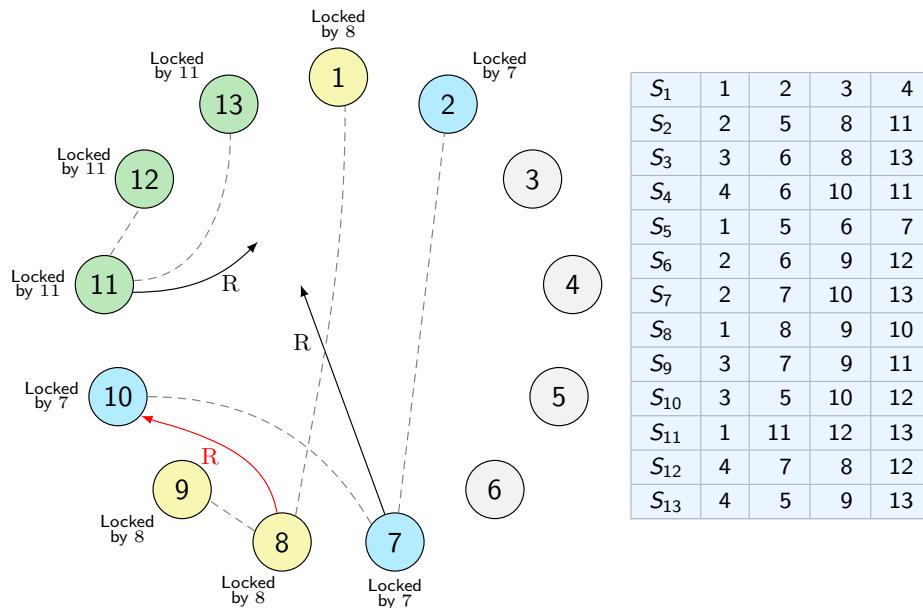
© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10-9

© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10 - 10

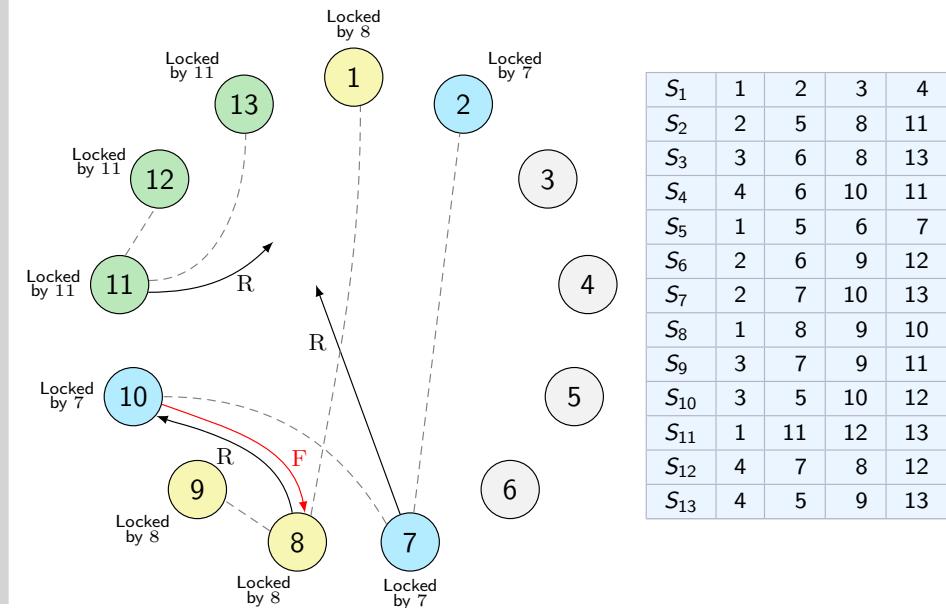
Beispiel



© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10–11

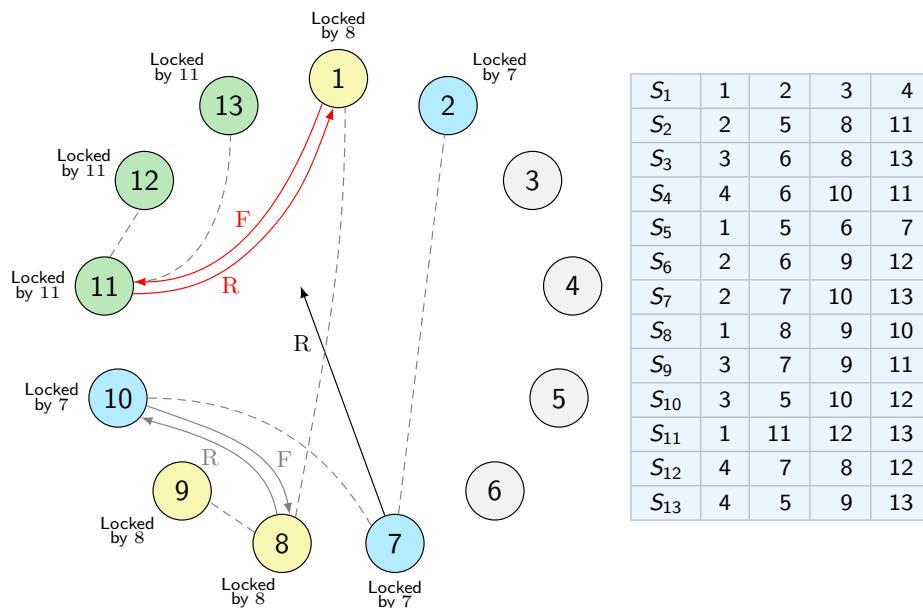
Beispiel



© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10–11

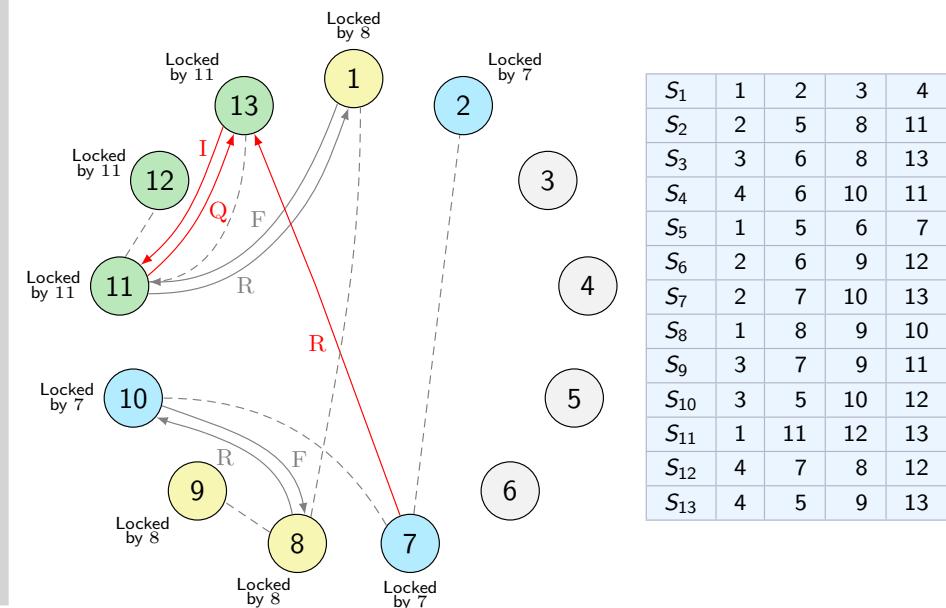
Beispiel



© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10–11

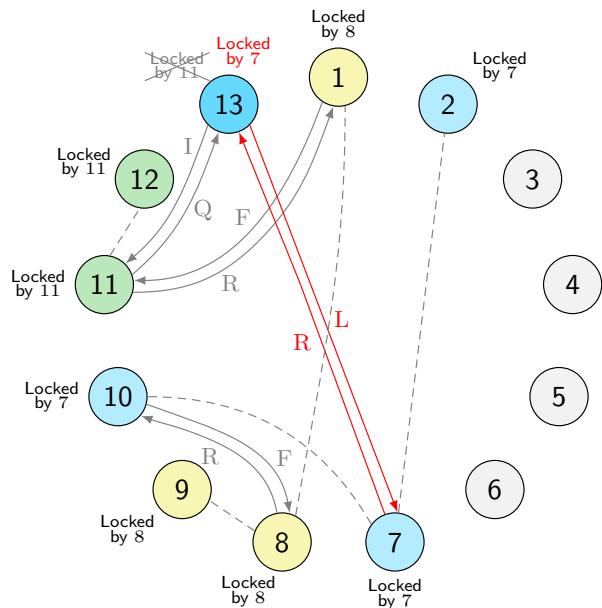
Beispiel



© td VS (SS16) Gegenseitiger Ausschluss – Maekawa-Algorithmus

10–11

Beispiel



Beispiel

