

**Eigenschaften mobiler und
eingebetteter Systeme:**

Fehlertoleranz auf Systemebene – Konsens

Dr.-Ing. Matthias Werner

Dipl.-Inf. Jan Richling

Wintersemester 2001/2002

Systemebene

- Beschreibung auf einer Abstraktionsebene, auf der Komponenten relativ unabhängig sind.
- Interaktion erfolgt über Nachrichtenaustausch
- Wenn nichts anderes gesagt, wird Synchronität angenommen:
Es gibt also für Aktionen (Kommunikation, Berechnungen) eine zeitliche Schranke, bis zu der die Aktion im fehlerfreien Fall abgeschlossen ist

Konsens

Konsens erlaubt einer Menge von fehlerfreien Verarbeitungseinheiten den korrekten Wert eines Teils des Systemzustandes zu erfahren und gleichzeitig zu wissen, daß jede andere korrekte Verarbeitungseinheit das selbe Ergebnis erhält, unabhängig von den Aktionen der fehlerhaften Einheiten.

Wir betrachten vier Klassen von Konsensproblemen:

- Systemdiagnose
- Gruppenmitgliedschaft
- Voting
- Byzantinisches Agreement

Literatur

- D. K. Pradhan: Fault-Tolerant Computer System Design, Abschnitt 3.4 und Kapitel 8
- N.A. Lynch: Distributed Algorithms, Kapitel 6
- L. Lamport, R. Shostak, M. Pease: The Byzantine Generals Problem, Transaction of Programming Languages, Vol 4, Nr. 3, 1982
- M. Barborak, M. Malek, A. Dahbura: The Consensus Problem in Fault-Tolerant Computing, ACM Computing Survey Vol 25, Nr. 2, 1993 (Semesterapparat)

Systemdiagnose

Bei der *Systemdiagnose* (*system-level diagnosis* oder einfach *system diagnosis*) wird versucht, durch gegenseitiges Überprüfen von Verarbeitungseinheiten festzustellen, wer korrekt und wer fehlerhaft ist.

- Es existiere ein Test (Knoten A testet Knoten B) der eine Aussage „korrekt“ oder „fehlerhaft“ zurückgibt
- Nutzen: Als defekt erkannt Knoten sollen nicht mehr verwendet werden
- Problem: Falschaussagen defekter Knoten über andere Knoten
- Aufgabenstellungen
 - Kann eine Lösung existieren? (Charakterisierung)
 - Wie sieht die Lösung aus?

t-Diagnostizierbarkeit und Syndrom

- Ein System heißt *t*-diagnostizierbar, wenn für jede beliebige Verteilung von bis zu *t* Fehlern im System jeder dieser Fehler erkannt und lokalisiert werden kann.
- Annahme: Es gibt einen externen Beobachter, der die Ergebnisse der Tests „einsammeln“ und bewerten kann.
- Die Menge der Ergebnisse wird *Syndrom* genannt
- Ein System ist genau dann *t*-diagnostizierbar, wenn es zu jeder Fehlerverteilung mit bis zu *t* Fehlern unterscheidbare *Syndrome* gibt.

PMC-Modell

- PMC-Modell von PREPARATA, METZKE und CHIEN, 1967
- Folgende Annahmen über Testausgänge gelten:

Tester	Testkandidat	Testausgang
korrekt	korrekt	korrekt
korrekt	fehlerhaft	fehlerhaft
fehlerhaft	korrekt	unbestimmt
fehlerhaft	fehlerhaft	unbestimmt

- Zur Vereinfachung sei ein Testausgang „korrekt“ mit 0 bezeichnet und ein Testausgang „fehlerhaft“ mit 1

Diagnostizierbarkeit im PCM-Modell

- Unter der Bedingung, daß sich keine zwei Knoten jeweils gegenseitig testen, gilt

Theorem 11.1. *Ein System ist dann t -diagnostizierbar, wenn*

- $n \geq 2t + 1$
- *Jeder Knoten wird wenigstens von t anderen Knoten getestet*

- Allgemeiner Fall:

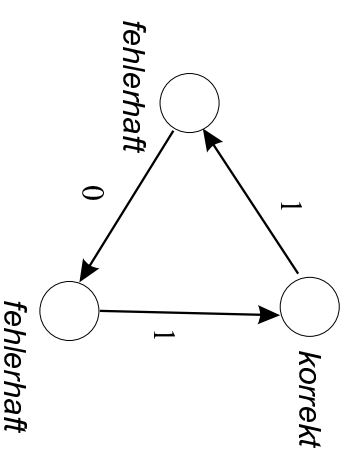
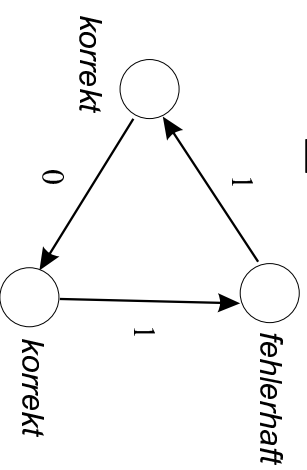
Theorem 11.2. *Ein System $G(V, N)$ ist genau dann t -diagnostizierbar, wenn*

- $n \geq 2t + 1$
 - $\forall v \in V : |\Gamma^{-1}(v)| \geq f$
 - $\forall p \in \mathbb{N}, 0 \leq p < f, \forall X \subseteq V, |X| = n - 2f + p \rightarrow |\Gamma(X)| > p$
- $\Gamma(Z)$: Menge der von den Knoten aus Z getesteten Knoten,
 - $\Gamma^{-1}(Z)$: Menge der Tester von Knoten aus Z

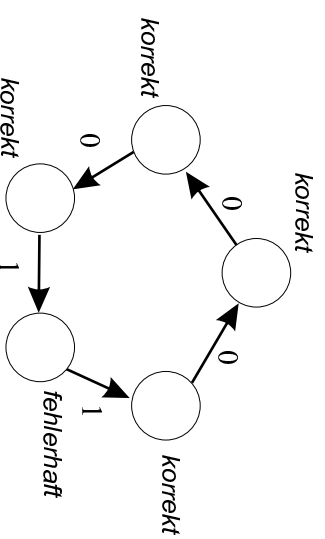
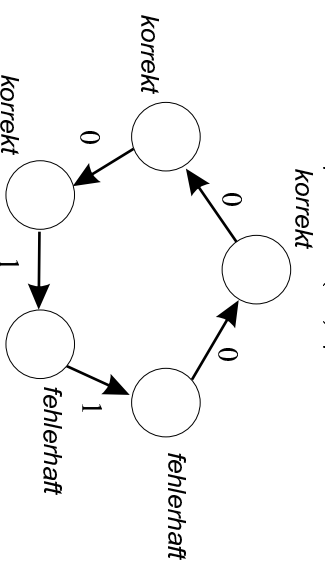
Beweis von Theorem 11.1

Notwendigkeit. Um die Notwendigkeit zu beweisen, reicht jeweils ein Gegenbeispiel.

- Notwendigkeit von $n \geq 2t + 1$:



- Notwendigkeit von $|\Gamma^{-1}(v)| \geq f$:

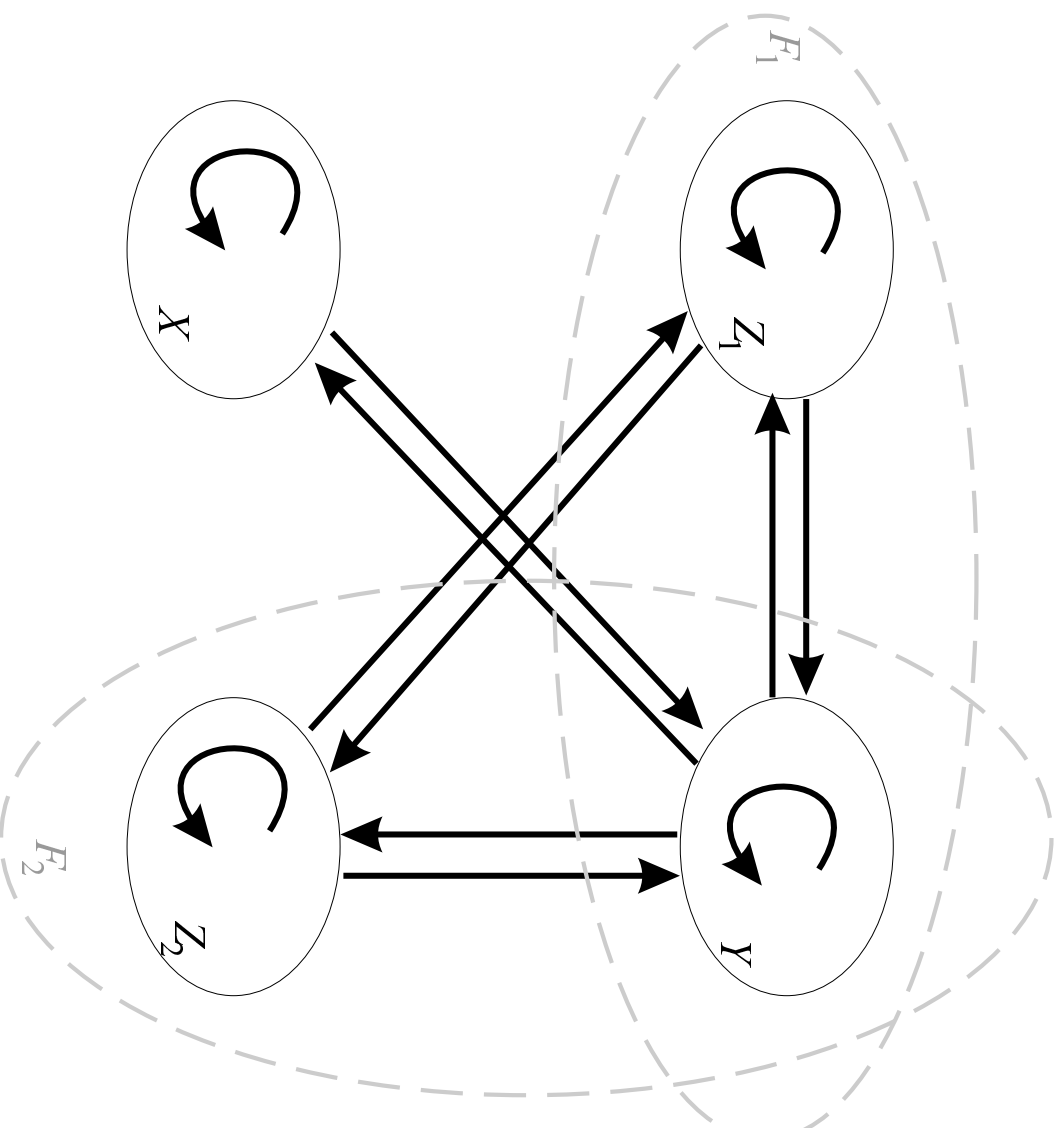


Beweis von Theorem 11.1 (II)

Hinlänglichkeit. Annahme des Gegenteils: S sei System mit $n \leq 2f$ und $|\Gamma^{-1}(v)| \geq f$.

- S sei nicht t -diagnostizierbar,
- \Rightarrow es gibt zwei Fehlermuster F_1 und F_2 , die das gleiche Syndrom S verursachen.
- Sei $Y = F_1 \cap F_2$, $Z_1 = F_1 - Y$ und $Z_2 = F_2 - Y$
- Sei X die Menge der in jedem Fall korrekten Elemente, $X = V - (F_1 \cup F_2)$
- Wenn F_1 und F_2 ununterscheidbar sein sollen, darf kein Element aus Z_1 oder Z_2 durch ein Element aus X getestet werden

Beweis von Theorem 11.1 (III)



Beweis von Theorem 11.1 (IV)

- Da keine gegenseitige Überprüfung: $|A + \Gamma^{-1}(A)| \geq |A| + t$
- Insbesondere: $|x + \Gamma^{-1}(x) + \Gamma^{-1}(\Gamma^{-1}(x))| \geq 2t + 1$
- $|F_1 \cup F_2| \leq 2t$
- D.h., jedes Element aus $F_1 \cup F_2$ oder mindestens einer seiner Tester muß von einem Element aus X getestet werden \Rightarrow Widerspruch

□

BGM-Modell

- PMC-Modell von BARSI, GRANDONI und MAESTRINI, 1976
- Folgende Annahmen über Testausgänge gelten:

Tester	Testkandidat	Testausgang
korrekt	korrekt	korrekt
korrekt	fehlerhaft	fehlerhaft
fehlerhaft	korrekt	unbestimmt
fehlerhaft	fehlerhaft	fehlerhaft

Diagnostizierbarkeit im BGM-Modell

Notwendige Bedingung für t -Diagnostizierbarkeit im BGM-Modell

Theorem 11.3. *Ein System S sei im BGM-Modell t -diagnostizierbar. Dann gilt:*

$$n \geq f + 2$$

Es gibt auch eine allgemeine Bedingung zur t -Diagnostizierbarkeit, aber die ist etwas komplizierter und wird hier ausgelassen.

Gruppenmitgliedschaft

- Gruppenmitgliedschaft (*group membership*) wird häufig als Spezialfall der Systemdiagnose betrachtet
- Problem: Ähnlich Systemdiagnose, aber Beschränkung auf Crash-Fehler

3 Ansätze nach CRISTIAN:

- Periodische Mitteilungen (*periodig broadcast*)
- Anwesenheitsliste (*attendance list*)
- Nachbarschaftsüberwachung (*neighbor surveillance*)

In der Praxis treten Gruppenmitgliedschaftsprotokolle meist nicht in ihrer reinen Form auf, sondern als *Gruppenkommunikationsprotokolle*, siehe Vorlesung 17.

Voting

- Mehrheitsabstimmung (*Voting*) ist die „klassischste“ Form des Konsens.
- Voting kommt z. T. innerhalb anderer Protokolle vor, z.B. in Protokollen zum Byzantinischen Agreement
- Tolerierte Fehler: Berechnungsfehler und eingeschlossene Fehlerklassen
- Häufig zentraler Beobachter \Rightarrow single point of failure
- m -aus- n -Voting: mindestens m von n Teilnehmern müssen zustimmen
- Varianten:
 - $m \geq \lfloor \frac{n}{2} \rfloor + 1$: absolute Mehrheit
 - $m < \frac{n}{2}$: relative Mehrheit
 - $m > \lfloor \frac{n}{2} \rfloor + 1$: qualifizierte Mehrheit

Zuverlässigkeit vs. Sicherheit

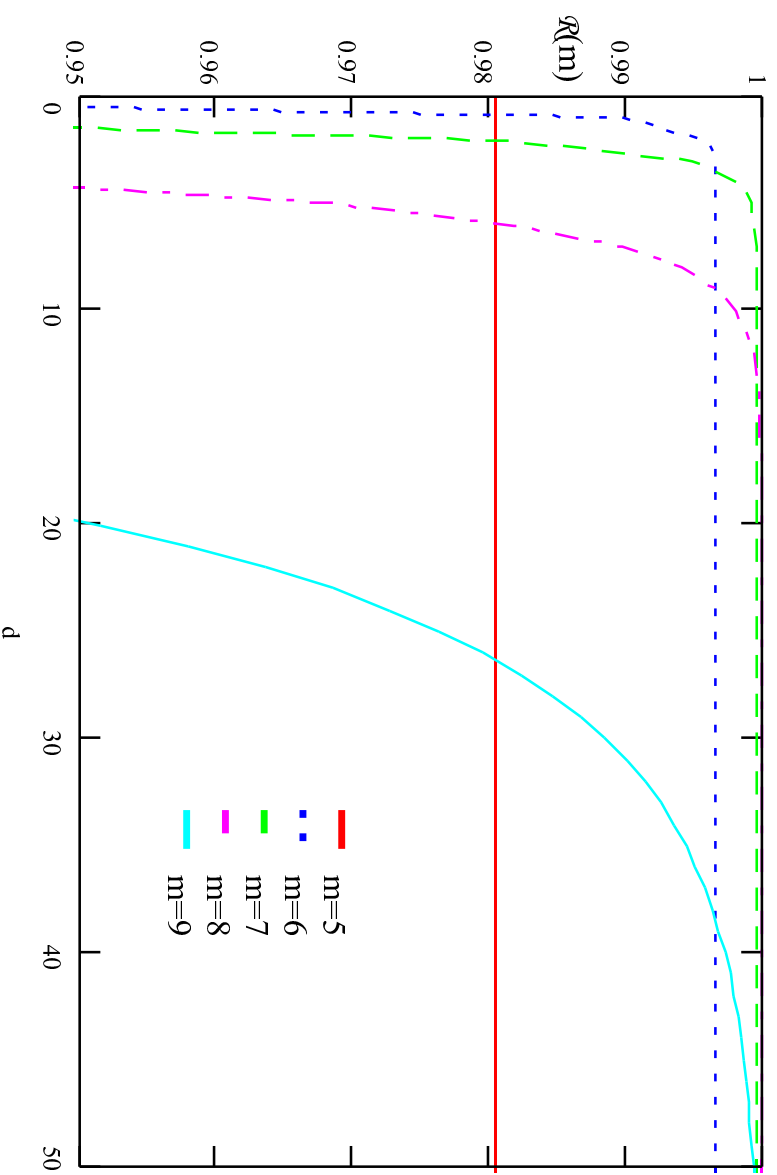
- Die Wahl von m beeinflusst die Zuverlässigkeit und die Sicherheit
- Je nach Anforderungen sollte unterschiedliche qualifizierte Mehrheit verlangt werden
- Zuverlässigkeit: $R_{ges}(R_0, n, m) = \sum_{i=m}^n \binom{n}{i} R_0^i (1 - R_0)^{n-i}$
- Sicherheit: $S_{ges}(R_0, n, m) = 1 - \sum_{i=m}^n \binom{n}{i} R_0^{n-1} (1 - R_0)^i$

Beispiel: m -aus-5 und $R_0 = 0.9$:

	Zuverlässigkeit	Sicherheit
3-aus-5	0.99144	0.99144
4-aus-5	0.91854	0.99954
5-aus-5	0.59049	0.99999

Responsives Voting

- In Abhängigkeit von der Deadline können unterschiedliche m optimal sein
- Beispiel: m -aus-9-Voting, $R_0 = 0.8$



Problem der Byzantinischen Generäle

- Byzantinisches Generalsproblem (auch interaktive Konsistenz) nach LAMPFORT
- Die Armee von Byzanz will mit mehreren Divisionen eine Stadt erobern
- Jede Division wird von einem General befehligt
- Generäle kommunizieren ausschließlich durch Boten
- Unter den Generälen können sich Verräter befinden
- Eroberung ist nur bei konsistentem Verhalten (der loyalen Generäle) erfolgreich
- **Problem:** Gesucht ist ein Verfahren (Algorithmus), das folgendes garantiert:
 - Alle (loyalen) Generäle kommen zur gleichen Entscheidung
 - Den Verrätern soll es nicht möglich sein, einen inkonsistenten oder schlechten Plan durchzusetzen

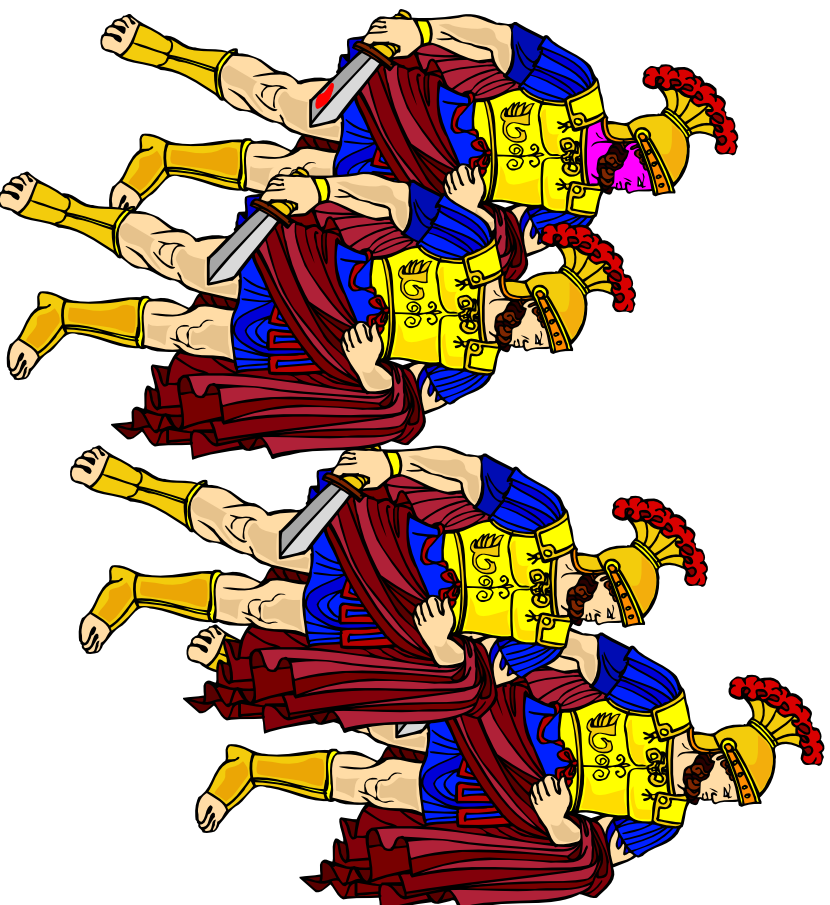
Problem der Byzantinischen Generäle (II)

Eine etwas formale Definition:

- Eine Menge von n Knoten v_i (Prozessoren, Rechnern, ...) sind durch ein (fehlerfreies) Netzwerk miteinander verbunden. Jeder Knoten besitzt einen zunächst privaten Wert δ_i aus einer Menge Δ möglicher Werte.
- Ziel ist es, daß jeder Knoten für einen Wert aus Δ entscheidet. Drei Bedingungen sollen erfüllt werden:
 - **Agreementbedingung.** Keine zwei (fehlerfreien) Knoten entscheiden sich für unterschiedliche Werte
 - **Validierungsbedingung.** Wenn alle fehlerfreien Knoten mit dem gleichen privaten Wert $\delta \in \Delta$ beginnen, dann muß dies auch der Ergebniswert sein
 - **Terminierungsbedingung.** Alle fehlerfreien Knoten entscheiden sich nach endlicher Zeit

Anzahl der „Verräter“

Theorem 11.4. Wenn f die maximale Anzahl fehlerhafter Knoten in einem System von n Knoten ist, gibt es keinen Algorithmus, der das Problem des Byzantinischen Agreements löst, solange $n \leq 3f$ gilt.

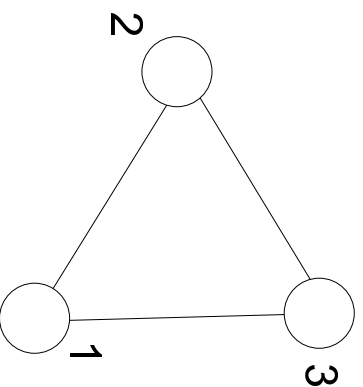


Beweis für Theorem 11.4

- Für $n = 2$: offensichtlich.
- Für $n > 2$: Beweisen zunächst einfacheres Lemma 11.4.1

Lemma 11.4.1. *Das Byzantinische Agreement ist nicht lösbar, wenn $n = 3$ gilt und mindestens ein Fehler möglich ist.*

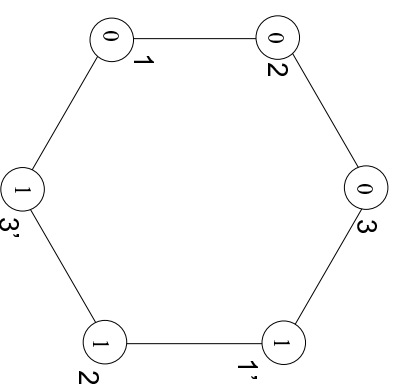
Betrachten System S mit 3 Knoten:



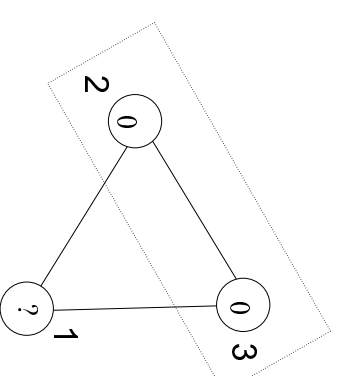
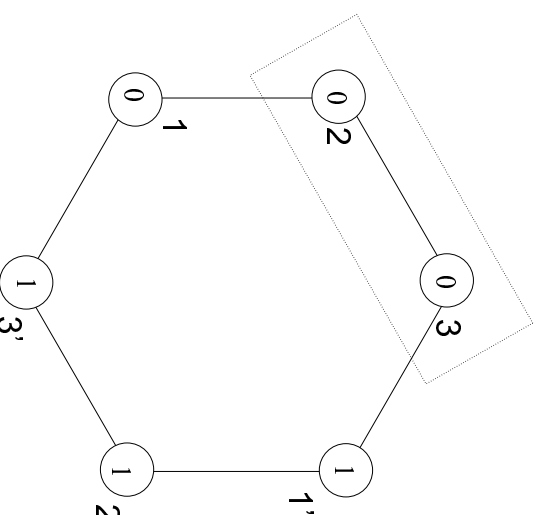
Annahme: In S lasse sich das Byzantinische Agreement lösen.

Beweis für Lemma 11.4.1

- Konstruieren System S' aus zwei S :

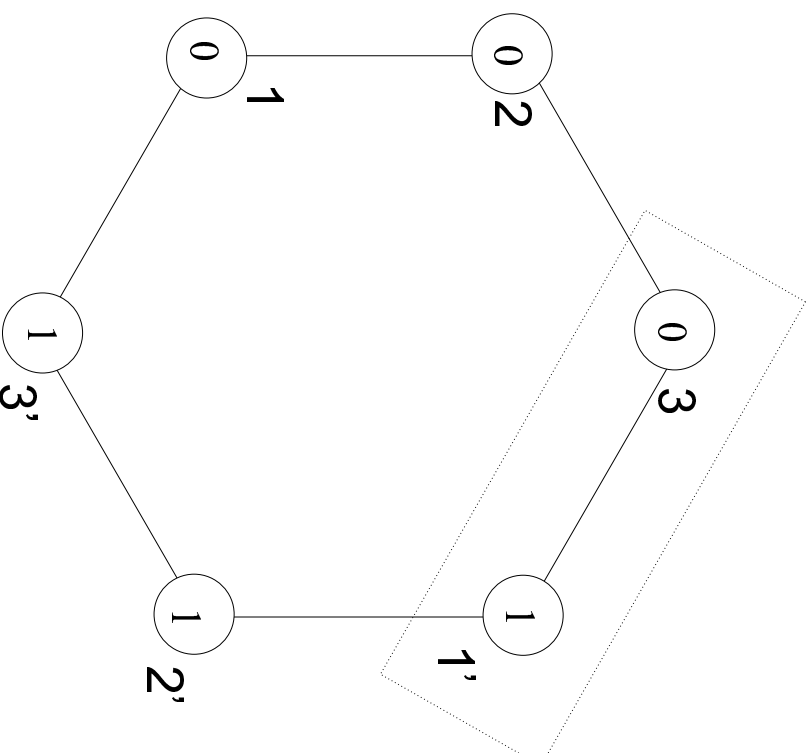


- Für zwei Knoten ist nicht der Unterschied zu S mit einem defektem Knoten erkennbar



Beweis für Lemma 11.4.1 (II)

Betrachtung jeweils zweier anderer Knoten führt zum Widerspruch



Beweis für Theorem 11.4 (Forts.)

- Für $n \leq 3f$: Annahme des Gegenteils, sei \mathcal{A} ein System mit n Knoten, von denen f fehlerhaft sind.
- \mathcal{A} löse das Agreementproblem.
- Partitioniere \mathcal{A} in drei nichtleere Teilmengen I_1, I_2 und I_3 mit $|I_i| \leq f$.
- OBdA: Alle fehlerhaften Knoten von \mathcal{A} sind entweder in I_1, I_2 oder I_3 .

Beweis für Theorem 11.4 (Forts.)

- Betrachten System \mathcal{B} , das durch \mathcal{A} simuliert wird.
- \mathcal{B} bestehe aus N_1 , N_2 und N_3 (entsprechend I_1 , I_2 und I_3)
- \mathcal{A} kann \mathcal{B} durch folgenden Spielregeln simulieren:
 - Alle Knoten in jedem I_i haben den gleichen Startwert v_i
 - Wenn alle Knoten in I_i sich für einen Endwert entscheidet, dann hat sich I_i (d.h. N_i) entschieden. Falls unterschiedliche Werte in I_i entschieden werden, wird einer davon genommen.
- Wenn \mathcal{A} das Agreement löst, wird das Agreement auch für \mathcal{B} gelöst \Rightarrow Widerspruch zu Lemma 11.4.1

□

Ein Algorithmus für Byzantinisches Agreement

- Sei $\Delta = \{0, 1\}$
- Sei $n > 3 \cdot f$
- Vollständig vernetzter Graph
- Algorithmus:
 - Runde 1: Jeder Knoten sendet jedem seinen Wert
 - Runde i : Jeder Knoten sendet jedem alle Werte, die ihm andere Knoten mitgeteilt haben (strukturfalsche Nachrichten werden ignoriert)
 - Jeder Knoten baut sich einen Baum aus den mitgeteilten Werten
 - Nach $n + 1$ Runden werden beim Baum jeder Verzweigung der Mehrheitswert der Kinder zugewiesen. Der Wert der Wurzel ist der Entscheidungswert.

Bedingungen für Byzantinisches Agreement

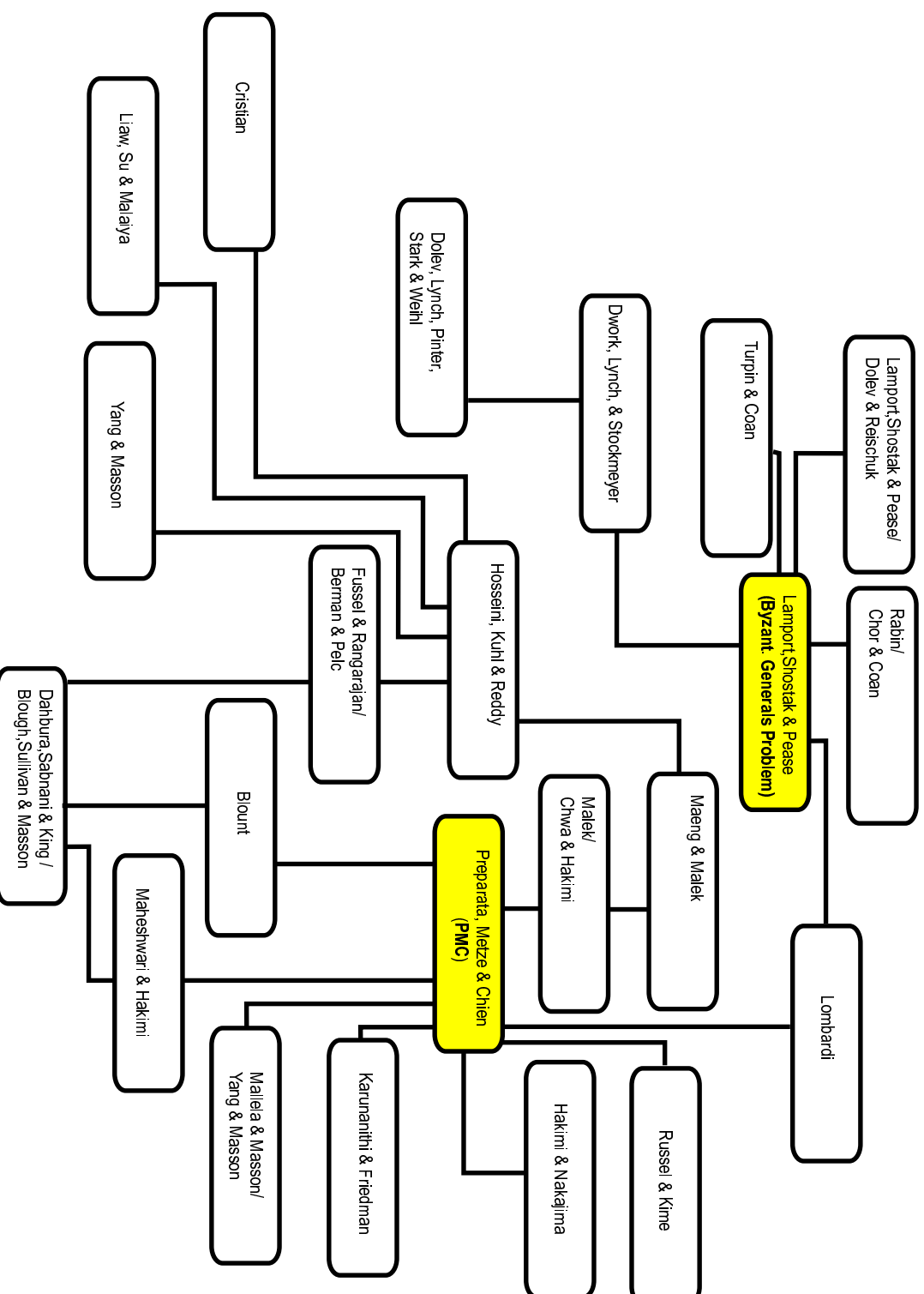
- Es gibt ziemlich viel Unmöglichkeitsbeweise für das Byzantinische Agreement
- DOLEV ET AL. haben fünf Systemcharakteristika genannt, die eine Rolle spielen:
 - **Ausführung auf den Knoten:** synchron vs. asynchron
 - **Kommunikation:** synchron vs. asynchron
 - **Nachrichtenreihenfolge:** geordnet vs. ungeordnet
 - **Übertragungsmechanismus:** broadcast vs. Punkt-zu-Punkt
 - **Senden/Empfangen-Atomarität:** atomar vs. nicht atomar

Bedingungen für Byzantinisches Agreement (II)

Es konnten genau vier Fälle identifiziert werden, bei denen es Byzantinisches Agreement mit f fehlerhaften Knoten geben kann:

- Synchroner Knoten und synchroner Kommunikation
- Synchroner Knoten und geordnete Nachrichtenreihenfolge
- Broadcast-Übertragung und geordnete Nachrichtenreihenfolge
- Synchroner Kommunikation, Broadcast-Übertragung, atomares Senden/Empfangen

Verwandschaft zwischen Konsensproblemen



Anwendungen

- Konsensprobleme kommen in einer Vielzahl von Anwendungen vor.

