Es genügt zu zeigen, daß $V_i = V_j$ ist, wenn die Prozessoren i und j fehlerfrei sind.

Prozessor i nehme in Schritt 2 v in V auf.

Macht er dies in Schritt 2A, dann sendet er ihn auch an j.

Macht er dies in Schritt 2B, dann muß er eine Nachricht $v{:}0{:}j_1{:}\ldots{:}j_k$ erhalten haben.

Es sind zwei Unterfälle zu betrachten:

1. $k < m$: **In diesem Fall sendet er die Nachricht $v{:}0{:}j_1{:}\ldots{:}j_k{:}i$ an j, so daß auch Prozessor j diesen Wert erhält.**
2. $k = m$: **Anführer fehlerhaft ---> höchstens $m-1$ der Prozessoren $j_1, \ldots, j_m$ fehlerhaft. Also muß wenigstens einer fehlerfrei sein und den Wert auch an Prozessor j gesendet haben.**

11.3.3 Byzantinische Verständigung (Dolev)

Reduktion der Nachrichtenzahl durch Erhöhung der maximalen Rundenzahl

Es sei $n = 3m + 1$:

Definitionen:

- **Runde**

  Algorithmus arbeitet in Runden

  $q_r = (q_r^1, \ldots, q_r^n)$ **Zustand am Anfang der r-ten Runde**

  q_r^i **enthält u. a. alle bis zu Beginn der r-ten Runde empfangenen Nachrichten**
- **Abarbeitung einer Runde in 2 Schritten:**
  1. **Aussenden von Nachrichten in Abhängigkeit vom Zustand am Anfang der Runde**
  2. **Empfang aller in der Runde gesandten Nachrichten und Ermittlung des Anfangszustandes für die nächste Runde**

- **Nachrichtenarten**

  ***-Nachricht, die als Wert des Anführers 1 zusichert.**

  k-Nachricht, die mitteilt, daß k überzeugt ist, eine zu recht bestehende *-Nachricht erhalten zu haben.
- **Lokale Zustände $q^j \in Q$ mit**

  $Q = \wp((\{*\} \cup \{1, \ldots, n\}) \times \{1, \ldots, n\}) \times \mathbb{N}$

  Paare (Nachrichtenkennung, Sender), Runde
- $LOW = m + 1$ **und** $HIGH = 2m + 1$

  Wenn eine bestimmte Aussage von wenigstens LOW Prozessen erhalten wurde, so muß sie von wenigstens einem korrekten gemacht worden sein.

  Wenn eine bestimmte Aussage von wenigstens HIGH Prozessen erhalten wurde, so muß sie von wenigstens LOW korrekten Prozessen gemacht worden sein.

- $W_x^j(q) = \{i \mid i \in \{1, \ldots, n\} \wedge (x, i) \in \pi_1(q^j)\}$

  Menge aller Prozessoren, die an j eine x-Nachricht gesandt haben ($x \in \{*, 1, \ldots, n\}$).
- ***j unterstützt i direkt*, wenn j von i eine *-Nachricht erhielt (d. h. $i \in W_*^j(q)$).**
- ***j unterstützt i indirekt*, wenn $|W_i^j(q)| \geq LOW$ ist, d. h. j weiß infolge der Mitteilung wenigstens eines korrekten Prozessors, daß i eine *-Nachricht erhalten hat.**
- **j *ist überzeugt*, daß i eine *-Nachricht ausgesandt hat, wenn $|W_i^j| \geq HIGH$ ist.**
- $C^j(q) = \{k \mid k \in \{1, \ldots, n\} \wedge k \neq s \wedge |W_k^j(q)| \geq HIGH\}$ **ist die Menge der *von j bestätigten Prozessoren* (ohne den Anführer s),**

  d. h. j ist zu Recht überzeugt, daß eine Mehrheit korrekter Prozessoren eine *-Nachricht versandt hat.

Einigungsprotokolle: Algorithmus von Dolev

- Ein Prozessor j ist im Zustand q_r ***aktiviert***, d. h. er sendet in der r-ten Runde an alle einschließlich sich selbst eine *-Nachricht,

  wenn er schon in einer früheren Runde aktiviert war, d. h. $j \in W_*^j(q_r)$ ist, (I1)

  oder wenn $\left|C^j(q_r)\right| \geq \mathrm{LOW} + \max(0, \lceil r/2 \rceil - 2)$ ist (I2)

  oder wenn $r = 2$ ist und er in der ersten Runde vom Anführer eine *-Nachricht erhielt, d. h. $s \in W_*^j(q_2)$ ist. (I3)

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

Einigungsprotokolle: Algorithmus von Dolev

❑ Algorithmus

1. In der ersten Runde sendet ein korrekter Anführer eine *-Nachricht an alle einschließlich sich selbst, falls sein Wert 1 ist.
2. In der k-ten Runde ($k > 1$) senden die Prozessoren an alle anderen (einschließlich sich selbst)
   - eine *-Nachricht, wenn sie aktiviert sind, (M1)
   - eine j-Nachricht, wenn sie j direkt unterstützen, (M2)

     (d. h. $j \in W_*^i(q)$)
   - eine j-Nachricht, wenn sie j indirekt unterstützen. (M3)

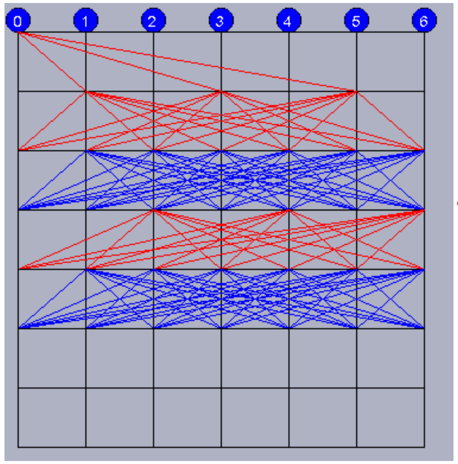
     (d. h. $\left|W_j^i(q)\right| \geq \mathrm{LOW}$)
3. Prozessoren, die von wenigstens HIGH Prozessoren überzeugt sind, d. h. $\left|\{i \mid \left|W_i^j\right| \geq \mathrm{HIGH}\}\right| \geq \mathrm{HIGH}$, entscheiden sich für den Wert 1.
4. Wenn ein Prozessor sich nach der $(2m+3)$-ten Runde für 1 entschieden hat, so nimmt er den Wert 1 als Ergebnis, sonst den Wert 0.

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

Einigungsprotokolle: Algorithmus von Dolev

Beispiel 6: Fehlerhafter Anführer ist P_0,
Teilnehmer P_2, P_4 und P_6 erhalten anfänglich keine Nachricht

q_1 q_2 q_3 q_4 q_5 q_6 q_7 q_8



*-Nachricht

k-Nachricht

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

Einigungsprotokolle: Algorithmus von Dolev

◆ Veranschaulichung

$i \rightarrow j \dashrightarrow k$: P_i hat an P_j eine *-Nachricht gesandt und P_j hat dies an P_k weitergemeldet

$k \in W_j^i$: $j \rightarrow k \dashrightarrow i$

$k \in W_*^i$: $k \rightarrow i$

$k \in C^i$: Es gibt mindestens m+1 korrekte vom Anführer verschiedene Prozessoren j_v mit $k \rightarrow j_v \dashrightarrow i$.
Prozessor k wird von i **bestätigt**.

i zeugt für j: $j \rightarrow i$

i zeugt indirekt für j: Für mindestens einen korrekten Prozessor k gilt $j \rightarrow k \dashrightarrow i$

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann
Reproduktion jeder Art oder Verwendung dieser Unterlage zu Lehrzwecken außerhalb der Universität Erlangen-Nürnberg ist ohne Genehmigung des Autors unzulässig

- **Aktivierung von i**

(I1) **i $\rightarrow$ i in vorangehender Runde**

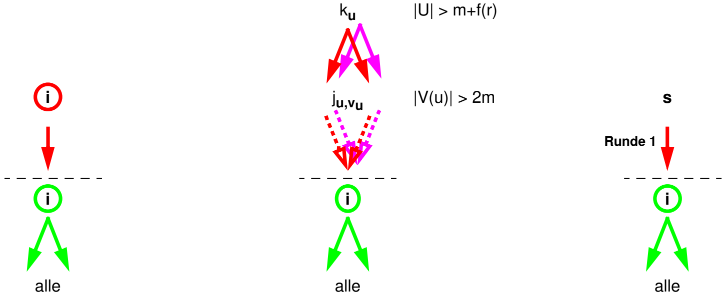
(I2) **$k_u \rightarrow j_{u,\,v_u} \dashrightarrow$ i mit** $u \in U \wedge |U| \geq m + 1 + f(r)$ **und** $v_u \in V_u \wedge |V_u| \geq 2m + 1$

(also $k_u \in C^i$**)**

(I3) **s $\rightarrow$ i in Runde 1**

- **Senden:**

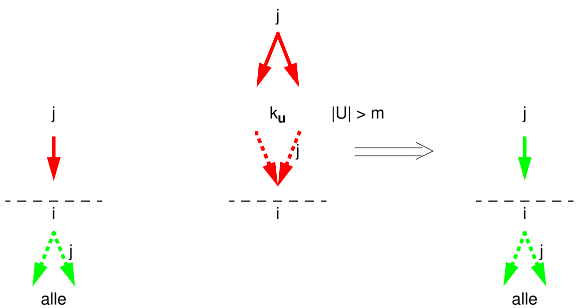
(M1) $(I1) \vee (I2) \vee (I3)$ **---> i $\rightarrow$ alle**



(I1)+(M1) (I2)+(M1) (I3)+(M1)

(M2) **j $\rightarrow$ i ---> j $\rightarrow$ i $\dashrightarrow$ alle**

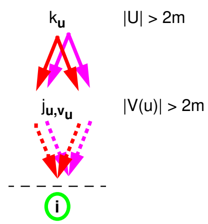
(M3) **Für alle** $u \in U$ **mit** $|U| \geq m + 1$**:** **j $\rightarrow$ k_u $\dashrightarrow$ i ---> j $\rightarrow$ i $\dashrightarrow$ alle**



(M2) (M3) = Transformation + (M2)

- **Entscheidung von i für Wert 1:**

$k_u \rightarrow j_{u,\,v_u} \dashrightarrow$ i mit $u \in U \wedge |U| \geq 2m + 1$ **und** $v_u \in V_u \wedge |V_u| \geq 2m + 1$



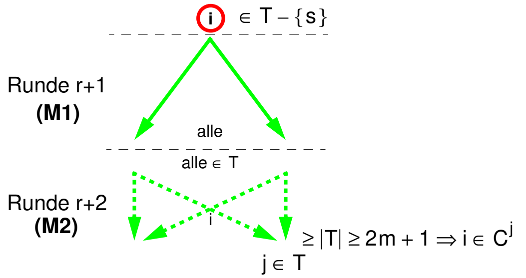
◆ **Beweis für das geforderte Verhalten**

L11.5 **Für korrekte Prozessoren sind die Mengen W und C mit der Rundenzahl monoton wachsend.**

Beweis: Definitionen

L11.6 $i, j \in T, i \neq s$: **i aktiviert in** q_r **---> i von j bestätigt in** q_{r+2}

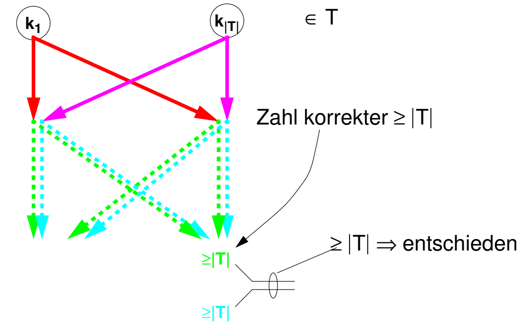
Beweis:



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.7 **Alle** $i \in T$ **aktiviert in** q_r **---> alle** $i \in T$ **entschieden in** q_{r+2}

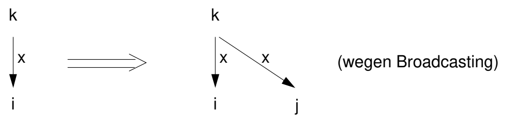
Beweis



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.8 $i, j \in T$: $\quad W_x^i(q_r) \cap T = W_x^j(q_r) \cap T$

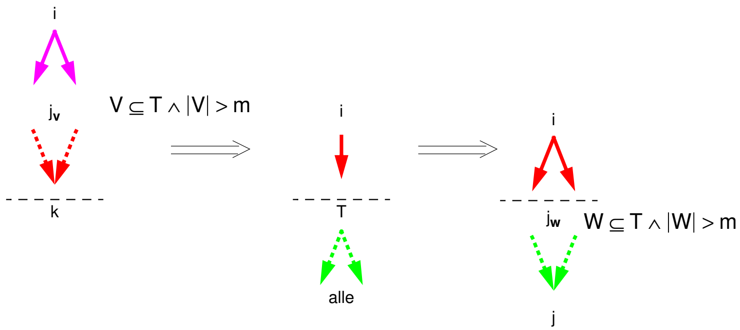
Beweis



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.9 $i, j \in T$: **ex.** $k \in T$ **mit** $i \in C^k(q_r)$ **---> für alle** $j \in T$ **ist** $i \in C^j(q_{r+1})$

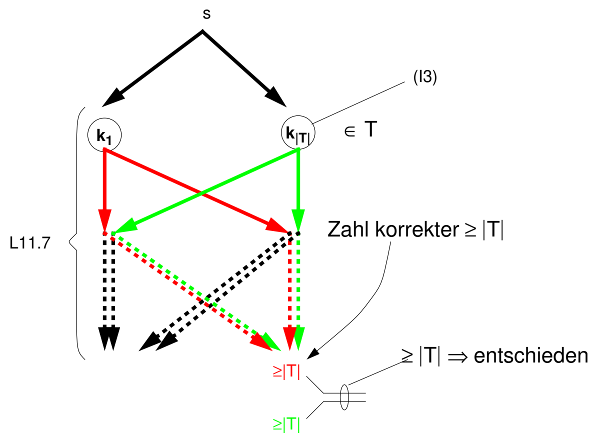
Beweis



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.10 $s \in T \wedge s_0 = 1$ **---> alle** $j \in T$ **entschieden in** q_4

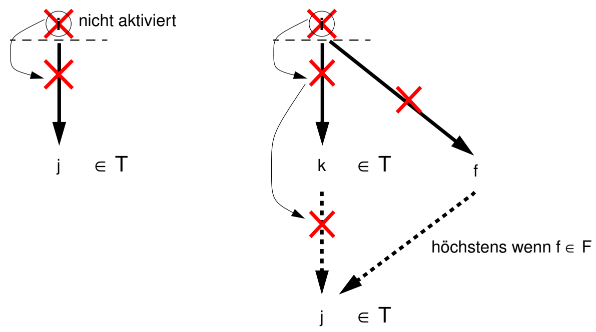
Beweis



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.11 $i, j \in T$**: i nicht aktiviert in** q_r **--->** $i \notin W^j_*(q_{r+1})$ **und** $|W^j_i(q_{r+2})| < LOW$

Beweis

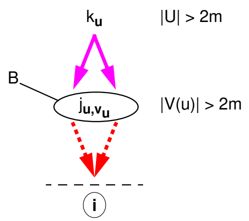


Wegen $|F| \leq m$ folgt die Beh.

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.12 $i \in T$**: i entschieden in** q_r **---> ex.** $B \subseteq T$ **mit** $|B| = LOW$ **und alle** $j \in B$ **aktiviert in** q_{r-1}

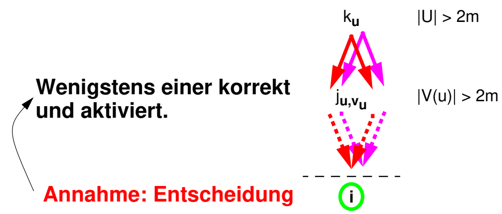
Beweis



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.13 $s \in T \wedge s_0 = 0$ **---> kein** $i \in T$ **(aktiviert oder entschieden in** q_{d+1}**)**

Beweis

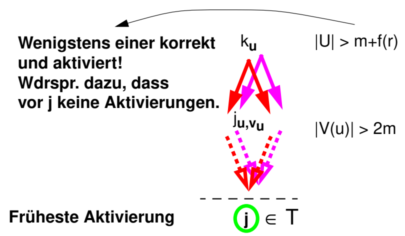


Also aktivierte Prozesse in Runde d.

Früheste Aktivierung erst in einer Runde >= 2, aktivierter Prozessor sei j.

Grund kann weder (I1) noch (I3) sein!

Also Aktivierung durch (I2).



Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.14 $A \subseteq T - \{s\} \wedge |A| = LOW$ **und alle** $i \in A$ **aktiviert in** q_r

---> alle $j \in T$ **entschieden in** q_{r+4}

Beweis

a) $s \in T$**: Wegen L11.13 muß** $s_0 = 1$ **sein. Dann folgt Behauptung aus (I3) und L11.7.**

b) $s \notin T$**: Es seien alle** $i \in A$ **aktiviert in** $q_{r'}$ **und r' minimal.**

Nach L11.6 alle $i \in A$ **von allen** $j \in T$ **bestätigt in** $q_{r'+2}$

b1) r' = 1: Kann nach Definition nicht sein.

b2) r' = 2: Wegen L11.6 alle $i \in A$ **von allen** $j \in T$ **bestätigt in** q_4**. Da f(4) = 0 also auch entschieden.**

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

b3) r' > 2: Es muß k existieren, das in $q_{r'}$ **erstmalig aktiviert wird. Grund kann nur I2 sein.**

$$\Rightarrow |C^k(q_{r'})| \geq LOW + \max(0, \lceil r'/2 \rceil - 2)$$

r' minimal $\Rightarrow k \notin C^k(q_{r'})$ **,** **denn** $k \in C^k(q_{r'}) \Rightarrow |W_k^k(q_{r'})| \geq HIGH$ **nach Def.**

$\Rightarrow k$ aktiviert in $q_{r'-2}$ **wegen L11.11, Wdrspr.**

$\forall j(j \in T \Rightarrow C^j(q_{r'+2}) \supseteq C^k(q_{r'}))$ **wegen L11.5 und L11.9.**

$\forall j(j \in T \Rightarrow k \in C^j(q_{r'+2}))$ **wegen L11.6.**

Wegen der letzten beiden Aussagen zusammen mit $k \notin C^k(q_{r'})$

$$\Rightarrow |C^j(q_{r'+2})| \geq LOW + \max(0, \left\lceil \frac{r'}{2} \right\rceil - 2) + 1 = LOW + \max(0, \left\lceil \frac{r'+2}{2} \right\rceil - 2)$$

also j aktiviert in $q_{r'+2}$ **wegen (I2) und wegen L11.6 entschieden in** $q_{r'+4}$**.**

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.15 $i \in T$**: i entschieden in** q_{d+1} **---> alle** $j \in T$ **entschieden in** q_{d+1}

Beweis

a) $(m = 0) \vee s \in T$ **:** **L11.10 und L11.13**

b) $m > 0 \wedge s \notin T$ **:** $\Rightarrow d > 4$

$i \in T$ **habe sich in** q_{d+1} **entschieden**

$\Rightarrow \exists A(A \subseteq T \wedge (|A| = LOW) \wedge \forall j(j \in A \Rightarrow j \text{ aktiviert\_in } q_d))$ **wegen L11.12.**

r sei erste Runde in der ein derartiges A existiert.

b1) $r \leq d - 3 \Rightarrow$ **Beh. wegen L11.14.**

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

b2) $r \geq d - 2 = 2m + 1 > 2$

$\Rightarrow \exists k(k \in A \wedge k \text{ aktiviert\_in } q_r \wedge \neg(k \text{ aktiviert\_in } q_{r-1}))$ **nach Def. von r und LOW**

Wegen (I2) $\Rightarrow |C^k(q_r)| \geq LOW + \max(0, \left\lceil \frac{r}{2} \right\rceil - 2) \geq LOW + m - 1 = HIGH - 1$

$s \notin T \wedge s \notin C^k(q_r)$ **nach Vorraussetzung und Def. von** $C^k(q_r)$

$$\Rightarrow \left| \overbrace{\{j | j \in N \wedge |W_j^k(q_r)| \geq HIGH\}}^{C^k(q_r)} \right| \geq HIGH - 1 \geq LOW, \textbf{ da } m \geq 1$$

$$\Rightarrow \left| \underbrace{\{k | k \in T \wedge |W_k^j(q_r)| \geq HIGH\}}_{\supseteq A'} \right| \geq LOW$$

$$\Rightarrow \exists A'(A' \subseteq T \cap C^k(q_r) \wedge (|A'| = LOW)$$
$$\wedge \forall j(j \in A' \Rightarrow j \text{ aktiviert\_in } q_{r-1})) \textbf{ wegen L7}$$

Widerspruch zur Wahl von r, also dieser Fall nicht möglich.

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

L11.16 $s \in T$: $s_0 = 1$ **---> alle** $j \in T$ **entschieden in** q_{d+1}

$s_0 = 0$ **---> kein** $j \in T$ **entschieden in** q_{d+1}

Beweis

1. Zeile folgt unmittelbar aus L11.7.

2. Zeile bewiesen als L11.13.

11.3.4 Algorithmus von Burns und Neiger (Einigungs-Algorithmus, 1994)

- **Bezeichnungen**

  n Zahl der Prozessoren

  m Maximalzahl fehlerhafter Prozessoren

  f Zahl tatsächlich fehlerhafter Prozessoren

- **Bedingungen**

  $f \le m$ **und** $n \ge m^2 + 4m + 1$

- **Partitionierung von** $m^2 + 4m + 1$ **Prozessoren in** $m + 1$ **disjunkte Gruppen** G_i

  G_1 **enthält** $2m + 1$ **Prozessoren.**

  Für $2 \le i \le m + 1$ **besteht** G_i **aus** $2(m + 2 - i) + 1$ **Prozessoren.**

  d. h. es enthält

  G_1 $2m + 1$ **Prozessoren,**

  G_2 $2m + 1$ **Prozessoren,**

  G_3 $2m - 1$ **Prozessoren,**

  .

  .

  .

  G_{m+1} 3 **Prozessoren.**

- **Algorithmus**

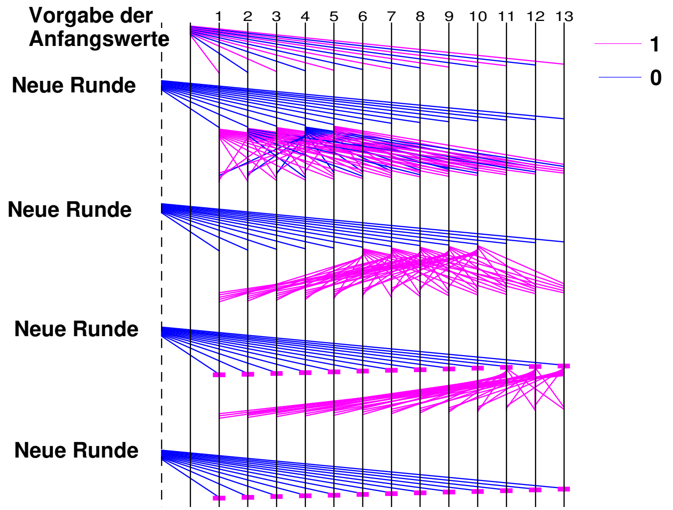
Der Algorithmus läuft in Runden.

- **Jeder Prozessor besitzt einen Wert, der anfänglich den Anfangswert darstellt.**
- **In jeder Runde r senden die Prozessoren der Gruppe G_r ihren Wert an alle anderen.**
- **Am Rundenende ermittelt jeder Prozessor der Runde seinen Wert durch Mehrheitsbildung. Für ausgebliebene Nachrichten wird der eigene Wert benutzt.**
- **Wenn ein Prozessor von allen Prozessoren der Gruppe r den gleichen Wert erhält, entscheidet er sich für diesen Wert.**
- **Prozessoren, die sich bis zum Ende der (t+1)-ten Runde noch nicht entschieden haben, entscheiden sich für ihren Wert.**

Jeder Prozessor p führt folgenden Algorithmus (in synchronisierten Runden) aus:

```
for (i = 1; i <= m + 1; i++) {
// Ein Schleifendurchlauf stellt jeweils eine Runde dar.
  if (G[i].contains(p))
    for (j = 1; j <= n; j++) send(j, p, v);
  tmp = 0;
  for (pix = G[i].first(); pix != 0; G[i].next(pix)) {
    receive(pix, &m); tmp += m;
  }
  if (!decided) {
    if (tmp == 0 || tmp == G[i].length()) {
      v = (tmp ? 1 : 0); decided = TRUE; break;
    } else {
      v = 2 * tmp > G[i].length() ? 1 : 0;
    }
  }
}
```


Beispiel: Einigungsprotokoll von Burns und Neiger (f = m = 2)



- **Grundgedanke**
  - **Da es $m + 1$ Gruppen gibt, muß wenigstens eine fehlerfreie existieren (d. h. alle ihre Mitglieder sind fehlerfrei).**

    Wenn eine fehlerfreie Gruppe sendet, erhalten alle die gleiche Multimenge von Werten.

    Wenn einmal alle fehlerfreien Prozessoren den gleichen Wert besitzen, so behalten sie ihn bei.
  - **Falls $f < m$ müssen unter $G_1, G_2, \ldots, G_{f+2}$ wenigstens zwei fehlerfrei sein.**
  - **Die Mitglieder der zweiten fehlerfreien Gruppe senden alle den gleichen Wert, also haben sich spätestens nach der Runde $f + 2$ alle fehlerfreien entschieden.**

(a) Wenn unter $G_1, \ldots, G_i$ wenigstens eine fehlerfreie Gruppe ist, beenden alle fehlerfreien Prozessoren die i-te Runde mit dem gleichen Wert.

(b) Wenn unter $G_1, \ldots, G_i$ wenigstens zwei fehlerfreie Gruppen sind, haben sich nach der i-ten Runde alle fehlerfreien Prozessoren entschieden.

(c) Wenn alle fehlerfreien Prozessoren, die i-te Runde mit dem gleichen Wert beginnen, beenden sie die Runde mit eben diesem Wert.

(d) Wenn sich ein fehlerfreier Prozessor am Ende der i-ten Runde entschieden hat, dann beenden alle fehlerfreien Prozessoren die Runde mit dem gleichen Wert.

Beweis durch vollständige Induktion

Induktionsanfang: i = 1

(a) G_1 fehlerfrei ---> alle erhalten gleiche Wertemenge
---> alle fehlerfreien entscheiden sich für gleichen Wert

(b) entfällt

(c) Folgt aus a) und Gewinnung des Wertes durch Mehrheitsbestimmung

(d) Entschieden ---> alle erhaltenen Werte gleich
---> von den fehlerfreien gleichen Wert erhalten
---> bilden die Mehrheit, da G_1 mindestens 2m+1 Prozessoren enthält
---> bei allen liefert die Mehrheit den gleichen Wert

Induktionsannahme: Aussage gültig bis einschließlich i

Induktionsschluß: Fall i+1

(a) 1. $G_1, \ldots, G_i$ fehlerhaft
---> G_{i+1} fehlerfrei
---> alle erhalten gleiche Wertemenge
---> alle fehlerfreien ermitteln gleichen Wert

2. Wenigstens eine Gruppe G_j fehlerfrei, j sei maximal
---> alle fehlerfreien haben am Ende der j-ten Runden den gleichen Wert
---> alle fehlerfreien haben am Ende der folgenden Runden den gleichen Wert (Eigenschaft c))

(b) 1. $G_1, \ldots, G_i$ enthalten nur eine fehlerfreie Gruppe
---> G_{i+1} fehlerfrei
---> alle erhalten in der (i+1)-ten Runde lauter gleiche Werte
---> alle fehlerfreien entscheiden sich für diesen Wert

2. Wenigstens zwei Gruppen aus $G_1, \ldots, G_i$ sind fehlerfrei
---> alle fehlerfreien haben sich schon zu Beginn der (i+1)-ten Runde entschieden

(c) 1. $G_1, \ldots, G_i$ enthalten wenigstens eine fehlerfreie
---> Behauptung wegen a) und Algorithmus

2. $G_1, \ldots, G_i$ fehlerhaft
---> G_{i+1} enthält höchstens $m - i < 2(m + 2 - (i + 1)) + 1$ fehlerhafte
---> die fehlerfreien sind in der Mehrzahl
---> alle fehlerfreien entscheiden sich für den gemeinsamen Anfangswert

(d) G_p entscheide sich am Ende der (i+1)ten Runde
---> G_p hat von allen aus G_{i+1} den gleichen Wert bekommen und unter $G_1, \ldots, G_i$ ist höchstens eine fehlerfreie Gruppe
---> G_{i+1} enthält höchstens $m - i + 1$ fehlerhafte Prozessoren

Da G_{i+1} insgesamt $2(m + 2 - (i + 1)) + 1 > 2(m + 1 - i)$ Prozessoren enthält, ist die Mehrheit fehlerfrei
---> alle fehlerfreien haben am Ende der Runde den gleichen Wert

S11.1 **Satz: Der Algorithmus von Burns und Neiger löst das Einigungsproblem. Spätestens nach** $\min(f + 2, m + 1)$ **Runden haben sich alle Prozessoren entschieden.**

Beweis:

Terminierung: trivial

Richtigkeit: Unmittelbare Folgerung aus c)

Einigung: Da es $m + 1$ **Gruppen gibt, aber höchstens m fehlerhafte Prozessoren, muß wenigstens eine Gruppe fehlerfrei sein. Damit folgt aus a) die Einigung spätestens in der (m+1)-ten Runde.**

Ist $f < m$**, so enthält** $G_1, \ldots, G_{f+2}$ **wenigstens zwei fehlerfreie Gruppen, woraus nach b) die Behauptung folgt.**

11.4 PAXOS

11.4.1 Aufgabenstellung

- **Modellannahmen**
  - **Prozessoren**
    - **Mögliches Fehlverhalten: Fail Stop**
      d. h. Prozessoren sind entweder *lebendig* oder *ausgefallen*
    - **Wiederanlauf erlaubt**
  - **Kommunikationssystem**
    - **vollständig vermascht**
    - **zuverlässige FIFO-Verbindungen**
- **Aufgabenstellung**
  - **Einigung**
    - **Wertevorrat für Anfangswerte beliebig**
    - **Einigung auf den Anfangswert eines der Prozessoren**

- **Synchronisation**
  - **Die Ausführung von Zustandsübergängen erfolgt atomar**
  - **L maximale Ausführungsdauer der (atomaren) Durchführung eines Zustandsübergangs**
  - **D maximale Übertragungsdauer von Nachrichten**
- **Architektur**

Multi-Paxos (Folge von Einigungsvorgängen)
Basic Paxos (ein einzelner Einigungsvorgang)
Wahl eines Anführers
Ausfallerkennung
Nachrichtensystem
Hardware

11.4.2 Ausfallerkennung

- **Systemstruktur beschrieben mit asynchron gekoppelten E/A-Automaten**