Was ist möglich und was nicht: Theoretische Grenzen der byzantischen Fehlertoleranz bei verteilten Anwendungen.

Mykhaylo Varshavskyy

simyvars@stud.informatik.uni-erlangen.de

Lehrstuhl für Informatik 4
Universität Erlangen-Nürnberg

Einleitung

- Grundlegendes Problem: Erreichen einer Einigung zwischen den beteiligten Prozessen.
- Beispiel: “transaction commit problem”.
  - Menge von Prozessen beteiligt sich in einer Transaktion.
  - Commit oder Rollback.
  - Alle Dateimanager sollen die gleiche Entscheidung treffen.
- Notwendigkeit eines fehlertoleranten Protokolls.
  - Reales System ist ein Gegenstand von verschiedenen Fehlern.

Modell

- Das Ausgangsmodell für die Einigung wird im folgenden beschrieben:
  - Insgesamt n Prozesse
  - Maximal m fehlerhafte Prozesse
  - Voll vermaschtes System
  - Zu jeder Nachricht sowohl Absender als auch Empfänger sind bekannt
  - Kommunikation ist zuverlässig
  - Abarbeitung synchron oder asynchron

Einigungsproblem

- Jeder von n Prozessen besizt einen Anfangswert (0 oder 1).
- Alle fehlerfreien Prozesse einigen sich auf den gleichen Wert.
- Wenn alle fehlerfreien Prozesse den gleichen Anfangswert haben, erfolgt die Einigung auf diesen Wert.
- send (p,m) - Übergabe der Nachricht m an das Kommunikationssystem für den Prozess p als Empfänger.
- recieve(p) - Entnahme von (p,m) oder leeres Ereignis.

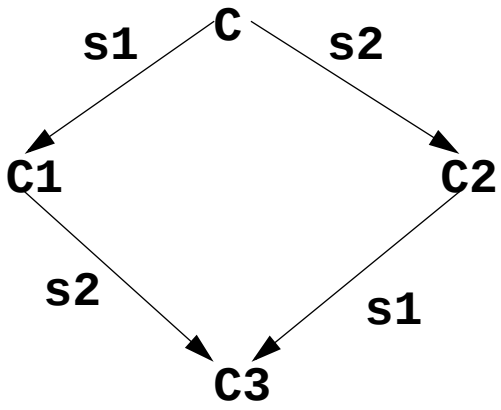
Bezeichnungen

- Eine Konfiguration beschreibt die Zustände aller Prozessoren und des Nachrichtensystens.
- Ein Schritt überführt eine Konfiguration in die andere durch Ausführung von recieve, send oder einen lokalen Übergang.
- Ein Übergang wird durch ein Ereignis e=(p,m) bestimmt. Eine Konfiguration C geht dabei in die Konfiguration e(C).
- Ablaufplan s ist eine Folge von Ereignissen die auf C angewandt werden kann.
- Lokales Ergebnis ist der Wert, wofür sich der Prozessor unwiderruflich entschieden hat.

Zum Beweis der Unlösbarkeitsaussage (1)

- Lemma1.

Ausgehend von der Konfiguration C führe s1 bzw. s2 zu C1 bzw. C2. Wenn die Prozesse, die in s1 einen Schritt ausführen, verschieden sind von denen, die in s2 eine Schritt ausführen, dann ist das Ergebnis von der Anwendung s2 auf C1 und s1 auf C2 gleich.



Zum Beweis der Unlösbarkeitsaussage (2)

- Eine Konfiguration C heisst bivalent, wenn die Menge lokaler Ergebnisse von C erreichbarer Konfigurationen enthält sowohl 0 als auch 1. Ansonsten heißt die Konfiguration 1- bzw.0-valent.

- Lemma2.

  Es gibt eine bivalente Anfangskonfiguration.

  - Beweis:
    - Annahme: Es gibt nur 0- und 1-valente Anfangskonfigurationen
    - Sei C0 0-valent, sei C1 1-valent, C0 und C1 unterscheiden sich im Wert des Eingaberegisters des Prozesses P.
    - s - zulässiger Lauf ohne Mitwirkung von P.
    - s kann sowohl auf C0 als auch auf C1 angewandt werden.

Zum Beweis der Unlösbarkeitsaussage (3)

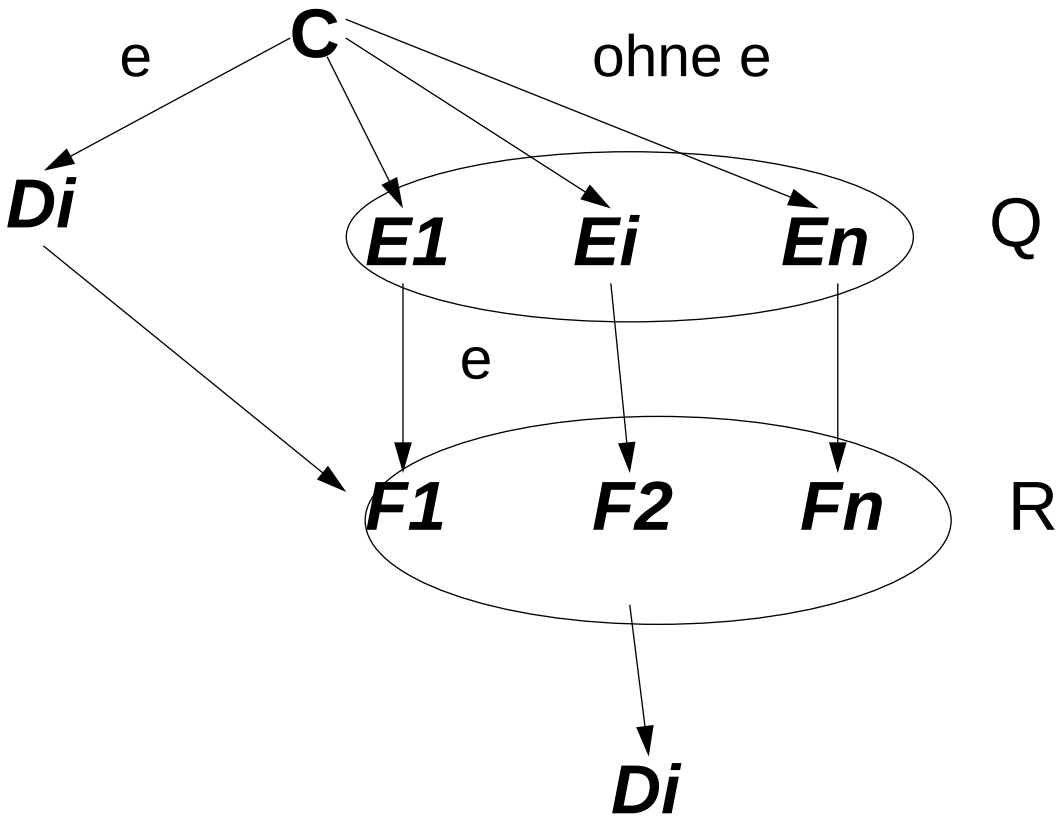
- s(C1) und s(C0) ermitteln die gleichen Entscheidungswerte. Widerspruch.

- Lemma 3

  Sei C eine bivalente Konfiguration und e=(p,m) ein in dieser Konfiguration anwendbares Ereignis.Sei Q die Menge von Konfigurationen, die von C aus ohne e erreicht werden können, und sei R=e(Q). Dann enthält R eine bivalente Konfiguration.

  - Beweis:
    - Annahme: R erhalte keine bivalente Konfiguration.
    - Di - von C aus erreichbare i-valente Konfiguration.
    - Entweder Di aus Q und Fi=e(Di) oder Di=e(C).
    - Di aus R oder es gibt Fi aus R, das von Ei erreichbar ist.
    - R muss 0- und 1-valente Konfigurationen enthalten.

Zum Beweis der Unlösbarkeitsaussage (4)



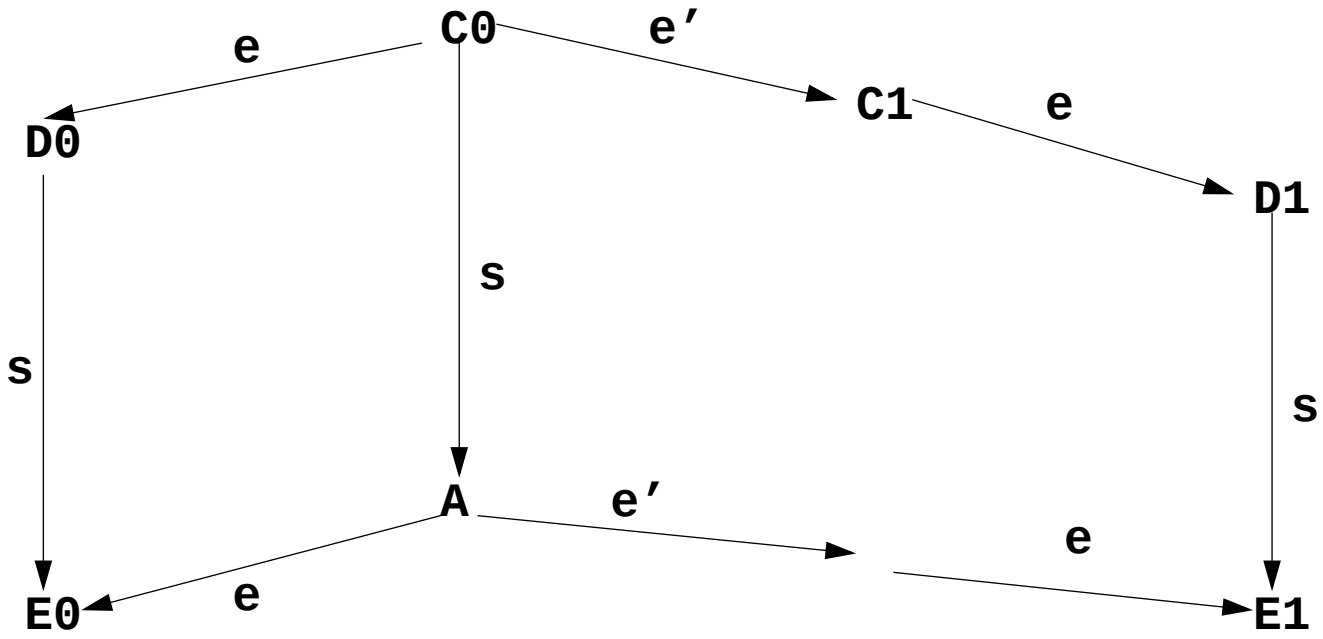
- Satz

Es existiert kein Protokoll, das unter Verwendung asynchroner Nachrichten das Einigungsproblem lösen kann, wenn nur ein Prozess fehlerhaft ist.

Zum Beweis der Unlösbarkeitsaussage (5)

- Annahme: Es existiert ein Protokoll, das auch bei einem fehlerhaften Prozess das Einigungsproblem löst.
- Wenn eine Konfiguration in einem Schritt in eine andere überführt werden kann, dann heißen die Konfigurationen Nachbarn.
- Seien C0 und C1 Nachbarn, so dass Di=e(Ci) i-valent ist. Sei C1=e'(C0) mit e'=(p',m').
- Wenn p'und p verschieden sind, dann gilt nach Lemma1 D1=e'(D0), aber dass ist unmöglich,denn ein Nachfolger einer 0-valententen Konfiguration kann nicht 1-valent sein.
- Wenn p=p', dann sei s ein Ablaufplan, wo sich der Prozess P nicht beteiligt und A=s(C0). s kann auch auf Di angewendet werden: Ei=s(Di) - die entsprechende i-valente Konfiguration.

Zum Beweis der Unlösbarkeitsaussage (6)



- Wegen e(A)=E0 und e(e'(A))=E1 muss A bivalent sein, aber es widerspricht der Annahme, dass A ein bis zur Entscheidung geführter Lauf ist.
- Somit ist die Annahme falsch.

Beweis der Unlösbarkeitsaussage (1)

- Beweis des Satzes erfolgt durch die Konstruktion eines nicht-entscheidenden, beliebig langen Laufs.
- Alle Prozesse werden in einer Warteschlange geführt, Nachrichten werden in einem Nachrichtenpuffer gesammelt.
- Ein Abschnitt besteht aus einem oder mehreren Schritten. Er endet, wenn der erste Prozess der Warteschlange einen Schritt ausgeführt hat, indem die älteste an ihn gerichtete Nachricht angenommen wurde.
- Danach wird der Prozess ans Ende der Warteschlange verwiesen.

Beweis der Unlösbarkeitsaussage (2)

- In einer endlosen Folge von Abschnitten führt jeder Prozess unendlich viele Schritte aus und empfängt jede an ihn gerichtete Nachricht
- Sei C0 eine bivalente Ausgangskonfiguration, dann existieren die Abläufe derart, dass jeder Abschnitt mit einer bivalenten Konfiguration wieder endet.
- C sei eine bivalente Konfiguration und P der erste Prozess in der Warteschlange. m oder leere Nachricht sei die älteste Nachricht an P in C.
- Sei e=(p,m). Nach Lemma3 gibt es eine bivalente Konfiguration C', die von C aus m einem Ablaufplan erreichbar ist, in dem die Anwendung von e der lezte Schritt ist. Diese Folge definiert einen Abschnitt.

Beweis der Unlöösbarkeitsaussage (3)

- Da alle Abschnitte mit einer bivalenten Konfiguration enden, wird in dem so konstruierten Ablaufplan nie eine Entscheidung erreicht.

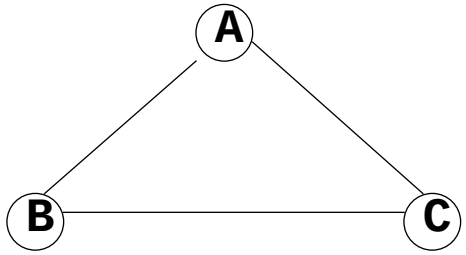
Unlösbarkeit bei zuvielen fehlerhaften Prozessen (1)

- Satz

In einem System mit n Prozessen, wovon maximal m fehlerhaft sind, existieren keine Protokolle zum Lösen des Bysantinischen Einigungsproblems, wenn n<=3m ist.

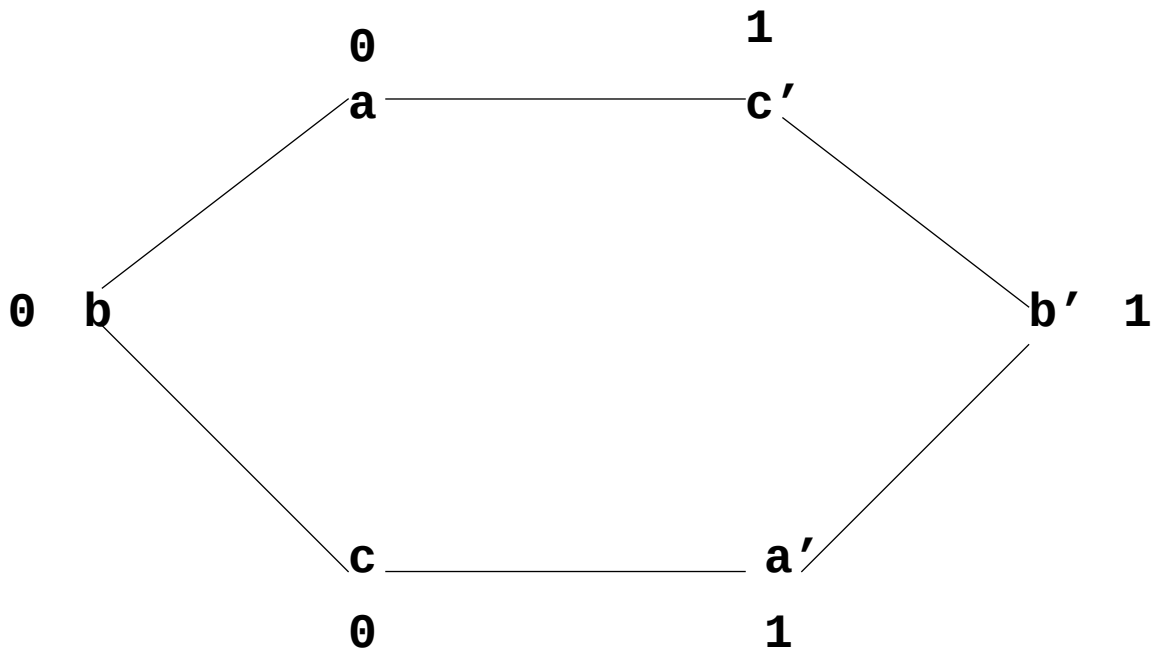
Beweis:

- Sei n=3 und m=1
- Annahme: Die Knoten A,B und C können sich im gezeichneten Graphen immer einigen.



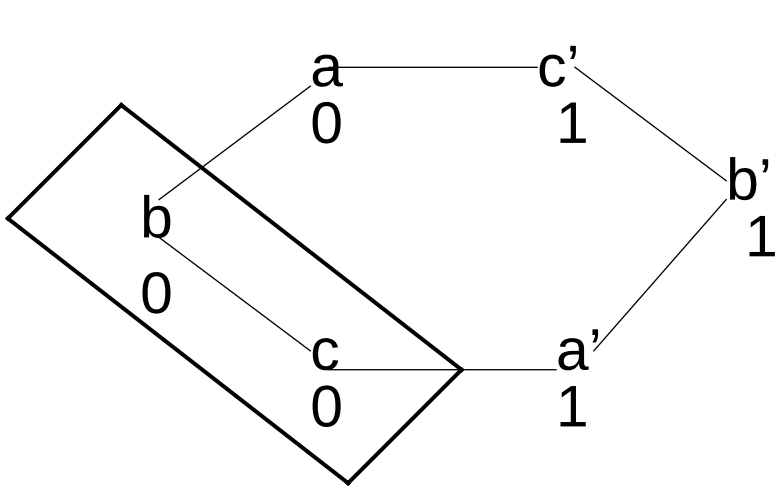
Unlösbarkeit bei zuvielen fehlerhaften Prozessen (2)

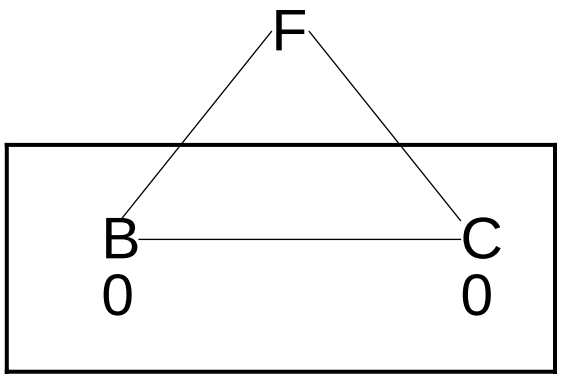
- In der Überlagerung S beschreibt das Gesamtverhalten dieses Systems. Die Knoten kommen hier zweimal vor: einmal mit dem Wert 0 und einmal mit dem Wert 1.



Unlösbarkeit bei zuvielen fehlerhaften Prozessen (3)

- Betrachten wir 3 Szenarien.
- Im ersten Szenario Knoten b und c sind fehlerfrei. Knoten A ist fehlerhaft.

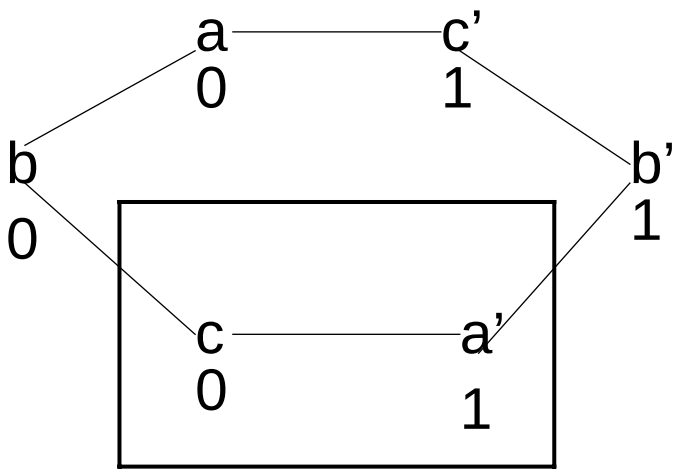


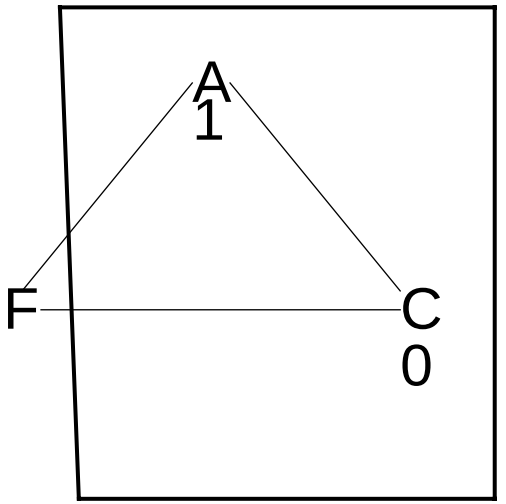


- b und c einigen sich auf den Wert 0.

Unlösbarkeit bei zuvielen fehlerhaften Prozessen (4)

- Im zweiten Szenario hat c den Wert 0 und a’ den Wert 1. Knoten B ist fehlerhaft.

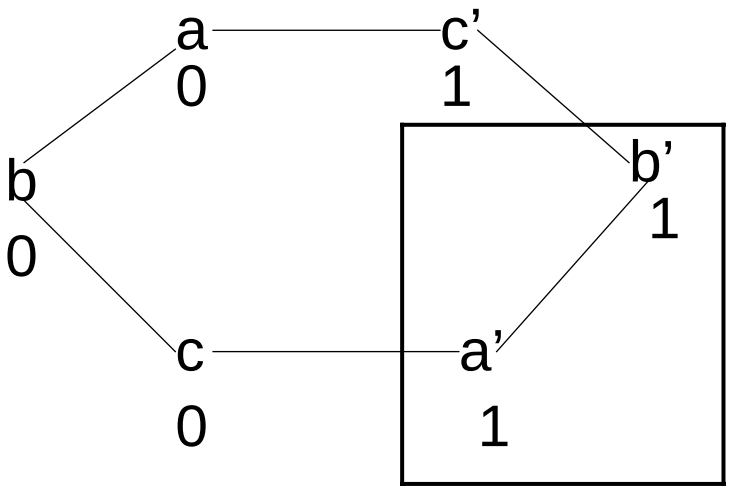


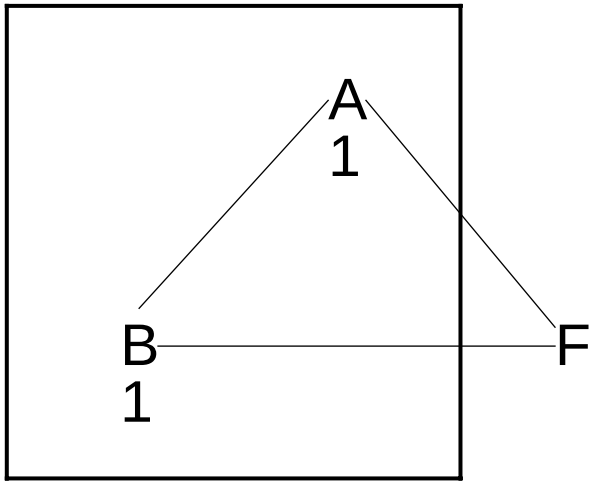


- Das Verhalten von C ist in den beiden Szenarien identisch. Deshalb kommt C zu den Ergebnis 0, dann soll auch A und a’ zum Wert 0 kommen.

Unlösbarkeit bei zuvielen fehlerhaften Prozessen (5)

- Im dritten Szenario hat a’ den Wert 1 und b’ - auch den Wert 1.
- Knoten C ist fehlerhaft.





- Knoten A zeigt dasselbe Verhalen wie im Szenario 2. Deshalb muss er sich für den Wert 0 entscheiden.

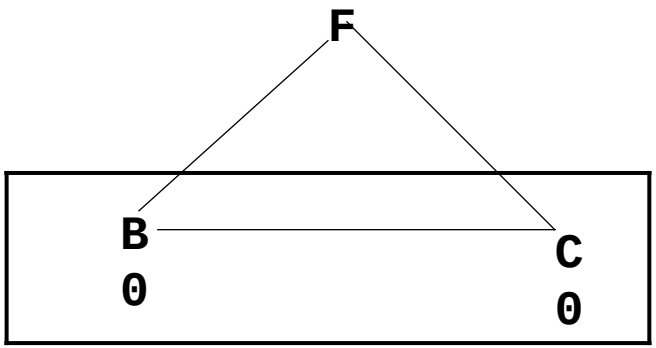
Unlösbarkeit bei zuvielen fehlerhaften Prozessen (6)

- Andererseits müssen sich Knoten a’ und b’ auf den Wert 1 einigen. Widerspruch.

Unlösbarkeit bei zuvielen fehlerhaften Prozessen (7)

- Wenn n<=3m ,dann wird die Knotenmenge in den Teilmengen a,b und c partioniert.
- Jede Teilmenge enthält höchstens m Elemente.
- Die gleiche Argumentation führt wieder zum Widerspruch.
- Ist Knoten A fehlerhaft, dann müssen B und C zusammen wenigstens n-m fehlerfreie Knoten enthalten, dann müssen sich alle korrekten Knoten in b und c für den Wert 0 entscheiden.

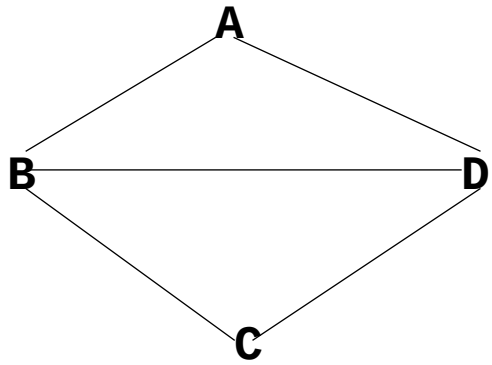
Unlösbarkeit bei zuvielen fehlerhaften Prozessen (8)



- So läßt sich die Argumentation analog zu dem Fall n=3 fortsetzen und führt wieder zum Widerspruch.

Weitere Unmöglichkeitsaussage (1)

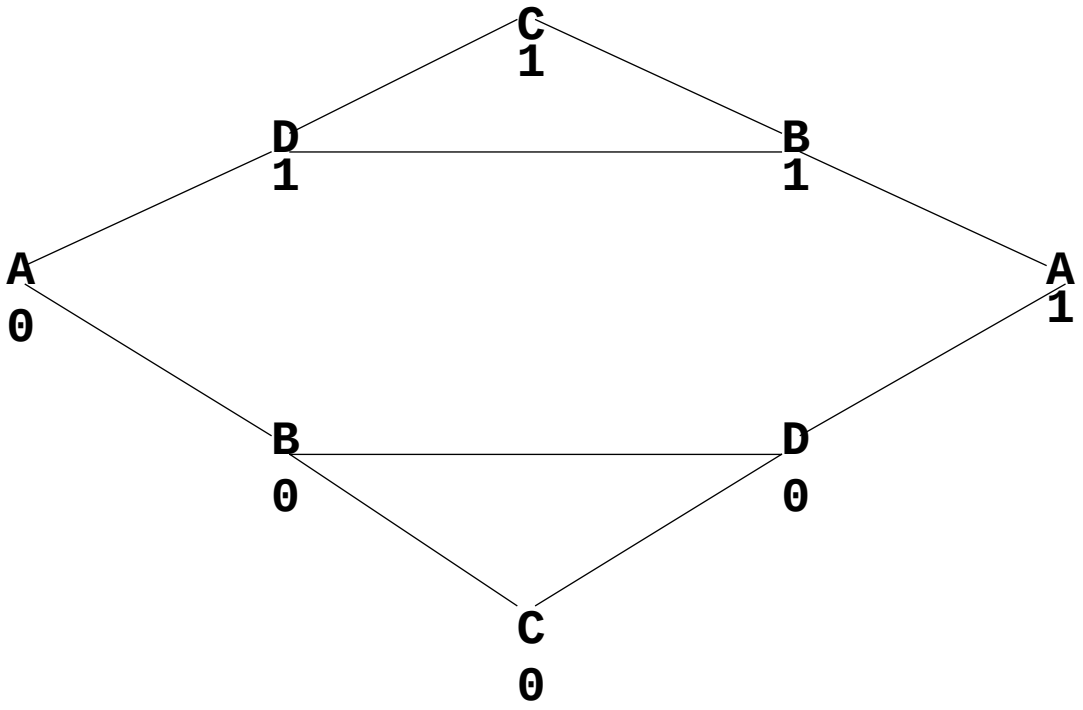
- Die Tolerierung von m möglichen Fehler erfordert die Verbundenheit von mindestens 2m+1.
  - Unter der Verbundenheit wird die zu entfernende Anzahl von Knoten verstanden, damit ein Graph in zwei Teile zerfällt.
  - Das Resultat wird zunächst für den Fall m=1 gezeigt.



  - Die Verbundenheit von G ist 2, denn B und D trennen G in zwei Teile: A und C.

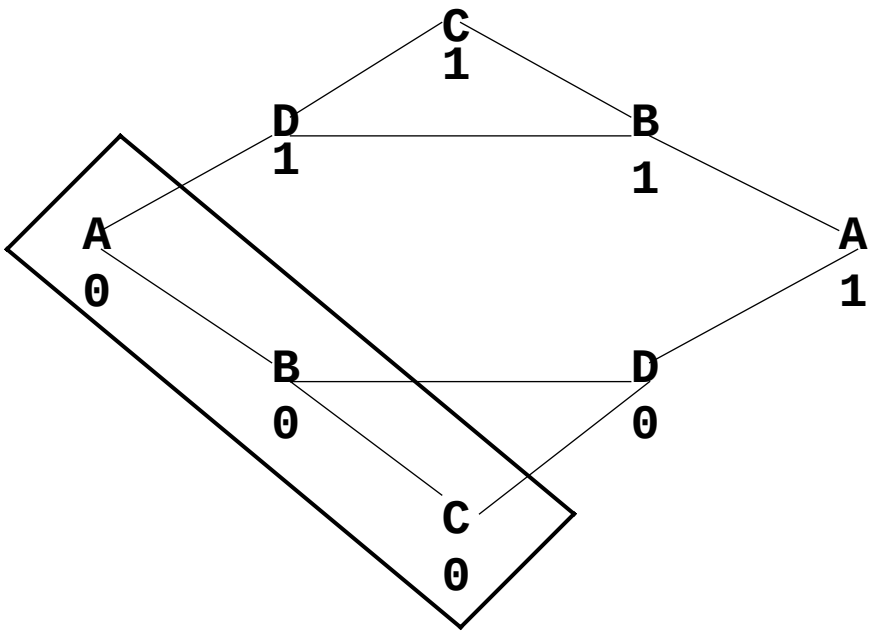
Weitere Unmöglichkeitsaussage (2)

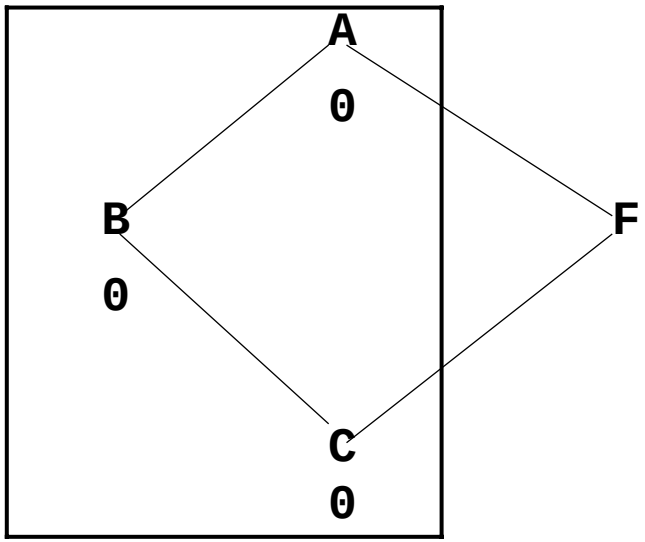
- Es wird das folgende System betrachtet: Graph S mit 8 Knoten



Weitere Unmöglichkeitsaussage (3)

- Das resultierende Verhalten des Systems sei V.Das Szenario V1 ist auf dem folgenden Bild ersichtlich.

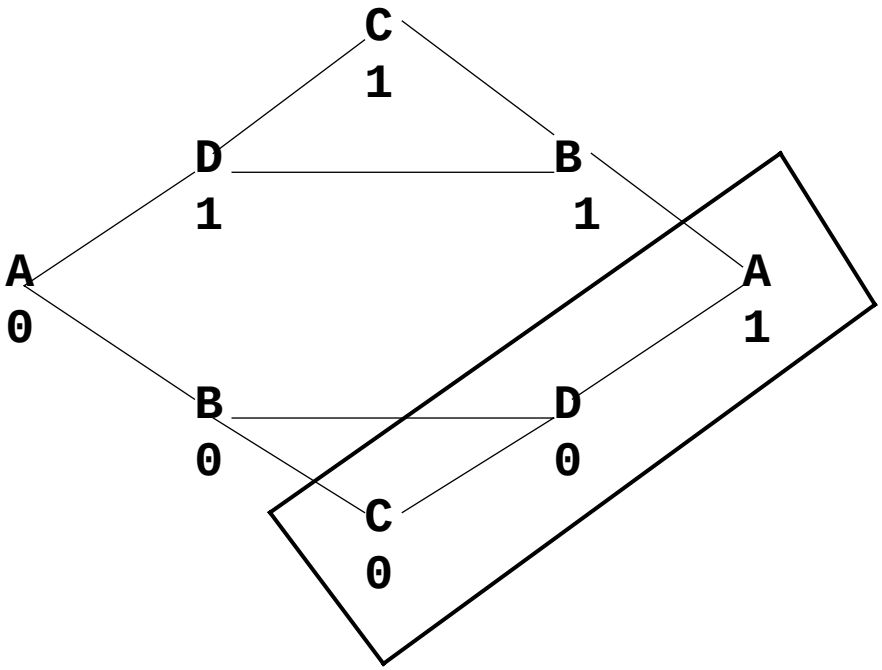


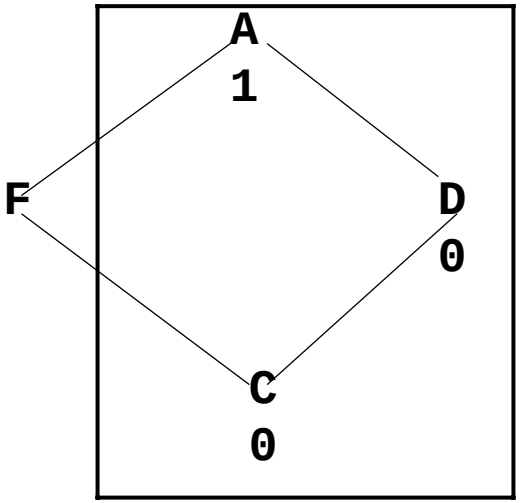


- A,B und C sind in dem Szenario V1 fehlerfrei. D ist fehlerhaft. D zeigt das eine Verhalten für A und das andere Verhalten für B und C.Dann müssen A, B und C zum 0 kommen.

Weitere Unmöglichkeitsaussage (4)

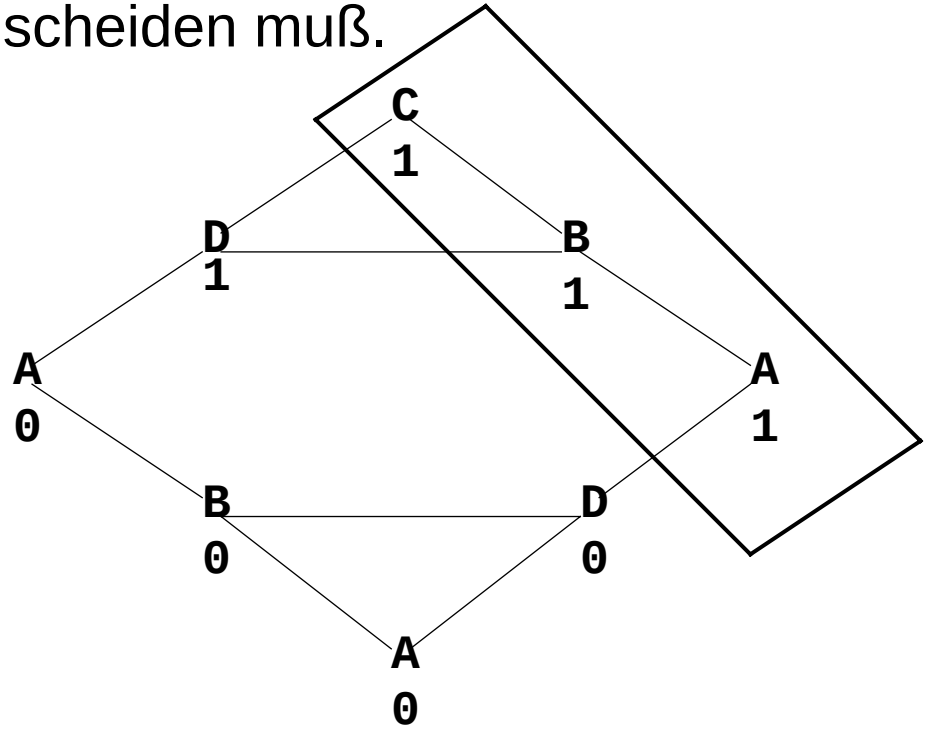
- Im zweiten Szenario ist B fehlerhaft.Die Einrichtung B zeigt ein Verhalten zu C und D und ein anderes zu A. Da C sich für 0 entschieden hat, müssen sich A,C und D auf 0 einigen.

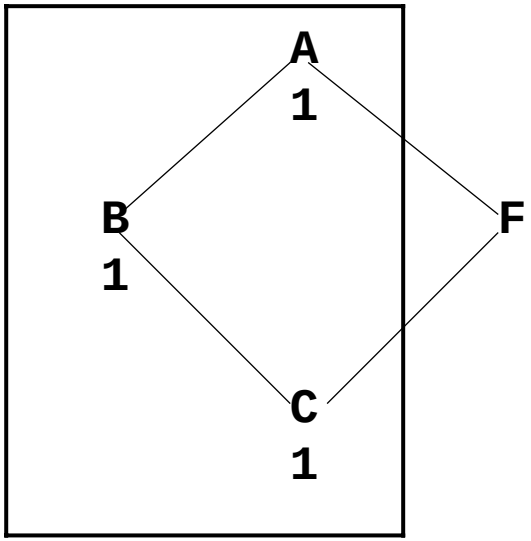




Weitere Unmöglichkeitsaussage (5)

- Im dritten Szenario sind A,B und C fehlerfrei, aber sie haben 1 als Eingabe. D ist hier fehlerhaft.Dann erfolgt Einigung auf dem Wert 1. Dies widerspricht der Tatsache, daß sich A für 0 enscheiden muß.





Weitere Unmöglichkeitsaussage (6)

Für den allgemeinen Fall könnnen die gleichen Bilder benutzt werden. Die B und D müssen dann je aus m Knoten bestehen, so dass die Entfernung von B und D zertrennt G in zwei nichtleere Mengen A und C.

- Für den allgemeinen Fall können die gleichen Bilder benutzt werden. Die B und D müssen dann je aus m Knoten bestehen, so dass die Entfernung von B und D zertrennt G in zwei nichtleere Mengen A und C.