

INTERNSCHRIFT Nr. 9

THEMA:

Untersuchung einer Koordinierungsmethode für zwei Prozesse

VERFASSER:

Bader

DATUM:

18.08.1969

FORM DER ABFASSUNG

ENTWURF

X AUSARBEITUNG

ENDFORM

SACHLICHE VERBINDLICHKEIT

X ALLGEMEINE INFORMATION
DISKUSSIONSGRUNDLAGE

ERARBEITETER VORSCHLAG

VERBINDLICHE MITTEILUNG

VERALTET

ÄNDERUNGSZUSTAND

BEZUG AUF BISHERIGE INTERNSCHRIFTEN

Vorkenntnisse aus:

Erweiterung von: Nr. 16 ("Semaphore")

Ersatz für:

BEZUG AUF KÜNFTIGE INTERNSCHRIFTEN

Vorkenntnisse zu:

Erweiterung in:

Ersetzt durch:

ANDERWEITIGE LITERATUR

Untersuchung einer Koordinierungsmethode für zwei Prozesse

1. Problemstellung

Diese Schrift ist der genaueren Untersuchung einer Koordinierungsmethode für zwei Prozesse gewidmet, die aus der in Internschrift Nr. 16 aufgezeigten DEKKER'schen durch Konstanthaltung der Größe "turn" hervorgeht. Beim TR 440 sind beide Methoden uninteressant, weil dort die entsprechende Koordinierung viel einfacher erreicht werden kann mit Hilfe des Befehls BL, der in einem einzigen Speicherzugriff den Inhalt einer Zelle ausliest und 0 in die Zelle einspeichert. Hier werden jedoch nur normale Lese- und Abspeicherungsbefehle vorausgesetzt und zwei Speicherzellen, auf die beide Prozesse zugreifen können.

Zunächst sei noch einmal kurz das Koordinierungsproblem umrissen: Es liegen zwei Prozesse vor, z.B. zwei Prozesse, die im Zeitmultiplex auf einem Rechnerkern laufen, oder zwei Prozesse, die auf verschiedenen Rechnerkernen laufen. In jedem Prozeß möchte man gewisse Abschnitte als "kritisch" auszeichnen können derart, daß immer gilt: Solange sich ein Prozeß in einem kritischen Abschnitt befindet, ist dem anderen Prozeß der Eintritt in einen kritischen Abschnitt verwehrt.

Dieses Koordinierungsproblem kann mit der hier beschriebenen Methode gelöst werden. Dabei wird folgende Behauptung bewiesen:

(1*) Zu keinem Zeitpunkt befinden sich beide Prozesse gleichzeitig in einem kritischen Abschnitt.

Unter der Voraussetzung, daß jeder Prozeß die nächsten paar Befehle immer in absehbarer Zeit ausführt, wird ferner gezeigt:

- (2*) Auf einen Zeitpunkt, in dem mindestens ein Prozeß eintrittswillig ist, kein Prozeß sich jedoch schon in einem kritischen Abschnitt befindet, folgt in absehbarer Zeit ein Eintritt in einen kritischen Abschnitt. (Mithin ist, wenn beide Prozesse gleichzeitig eintrittswillig sind, eine gegenseitige Blockierung ausgeschlossen.)

2. Beschreibung der Koordinierungsmethode

Die beiden Prozesse werden im folgenden A und B genannt. Es erweist sich als zweckmäßig, einen Prozeß in Bezug auf kritische Abschnitte in jedem Zeitpunkt durch genau eine der Eigenschaften

"uninteressiert",
 "sichtbar eintrittswillig",
 "unsichtbar eintrittswillig" und
 "in kritischem Abschnitt"

zu charakterisieren. (Die Eigenschaft "unsichtbar eintrittswillig" kommt nur bei Prozeß B vor.)

Zur Verständigung zwischen den beiden Prozessen werden zwei Speicherzellen verwendet, wobei die Schreibzugriffe auf die eine Zelle (Inhalt a) nur von Prozeß A her erfolgen, auf die andere Zelle (Inhalt b) ausschließlich von Prozeß B her. Die Größen a und b nehmen jeweils nur die Werte 0 und 1 an und haben folgende Bedeutung:

| Wert von a | Eigenschaft von Prozeß A |
|------------|--|
| 0 | "uninteressiert" |
| 1 | "sichtbar eintrittswillig" oder "in kritischem Abschnitt" |

| Wert von b | Eigenschaft von Prozeß B |
|------------|--|
| 0 | "uninteressiert" oder "unsichtbar eintrittswillig" |
| 1 | "sichtbar eintrittswillig" oder "in kritischem Abschnitt" |

(Anmerkung: Damit begrifflich der Inhalt einer Speicherzelle für alle Zeitpunkte festgelegt ist, wird für die Zeit zwischen zwei aufeinanderfolgenden Schreibzugriffen auf diese Zelle folgendes vereinbart: Wenn der erste Schreibzugriff im Zeitpunkt t_1 beginnt, und der darauffolgende im Zeitpunkt t_2 , so gelte für alle Zeitpunkte t mit $t_1 \leq t < t_2$ der beim ersten Zugriff eingespeicherte Wert.)

Die Koordinierung zwischen den Prozessen A und B wird nun durch das Einhalten folgender Regeln erreicht (vgl. Abb. 1):

1. Zu Beginn sind beide Prozesse "uninteressiert" (d.h. $a=b=0$).
2. Will Prozeß A in einen kritischen Abschnitt eintreten, so setzt er zunächst $a:=1$ und ist damit "sichtbar eintrittswillig". Dann liest er so oft b , bis er den Wert $b=0$ erhält: mit dem Zugriff auf $b=0$ beginnt der kritische Abschnitt. Am Ende des kritischen Abschnitts wird $a:=0$ gesetzt; damit ist Prozeß A wieder "uninteressiert".
3. Will Prozeß B in einen kritischen Abschnitt eintreten, so setzt er zunächst $b:=1$ und ist damit "sichtbar eintrittswillig". Darauf wird a gellesen: Mit dem Lesen von $a=0$ beginnt der kritische Abschnitt, der später durch $b:=0$ (d.h. Prozeß B ist "uninteressiert") endet; das Lesen von $a=1$ jedoch führt Prozeß B dazu, daß er dem Prozeß A gegenüber seinen Eintrittswunsch durch $b:=0$ verheimlicht ("unsichtbar eintrittswillig") und danach so oft a liest, bis der Wert $a=0$ (d.h. Prozeß A ist "uninteressiert") erhalten wird. Daraufhin wiederholt Prozeß B die obigen mit $b:=1$ beginnenden Aktionen.
4. Auf a und b finden nur die oben beschriebenen Zugriffe statt.

Im Ablaufplan für Prozeß A (s. Abb. 1) sind diejenigen Zeitpunkte, in denen von Prozeß A aus ein Zugriff auf a oder b beginnt, durch Kästchen dargestellt, die dazwischensliegenden Zeitintervalle durch gerichtete Linien. Die Zugriffe sind mit A1 bis A4 gekennzeichnet. Als "Zustand von Prozeß A" gilt immer die Kennzeichnung des letzten von Prozeß A begonnenen Zugriffs auf a oder b. (Als Anfangszustand von Prozeß A gilt A4.)

Der Ablaufplan für Prozeß B ist analog zu interpretieren.

3. Zustandsgraph

Um den zeitlichen Ablauf der beiden Prozesse gemeinsam verfolgen zu können, werden der jeweilige Zustand von Prozeß A und Prozeß B zusammengekommen betrachtet; als anfänglicher Zustand gilt dann (A4, B4). In Abb. 2 sind die Zustände als Punkte in einem Koordinatensystem dargestellt. Zustandsänderungen können nur auf Grund von Zugriffen auf a oder b vorkommen. Deshalb betrachtet man alle diejenigen Zeitpunkte, in denen ein Zugriff auf a, auf b, oder auf a und b gleichzeitig beginnt ("Zugriffszeitpunkte").

Jeder Zugriffszeitpunkt erscheint in Abb. 2 als eine Verbindungslinie, die vom ursprünglichen Zustand ausgeht (dickes Ende der Linie) und in den neuen Zustand (meist vom ursprünglichen verschieden) einmündet (dünnes Ende der Linie). Durch die Anfangsrichtung der Linie wird zum Ausdruck gebracht, von welchem Prozeß der Zugriff herrührt: waagrecht bei Prozeß A; senkrecht bei Prozeß B; diagonal, wenn beide Prozesse gleichzeitig mit einem Zugriff beginnen. (Da beide Prozesse nicht gleichzeitig auf dieselbe Zelle zugreifen können, kommt der letztere Fall nur dann vor, wenn ein Prozeß auf a und der andere auf b zugreift. Zustände, bei denen der nächste Zugriff von Prozeß A und Prozeß B auf

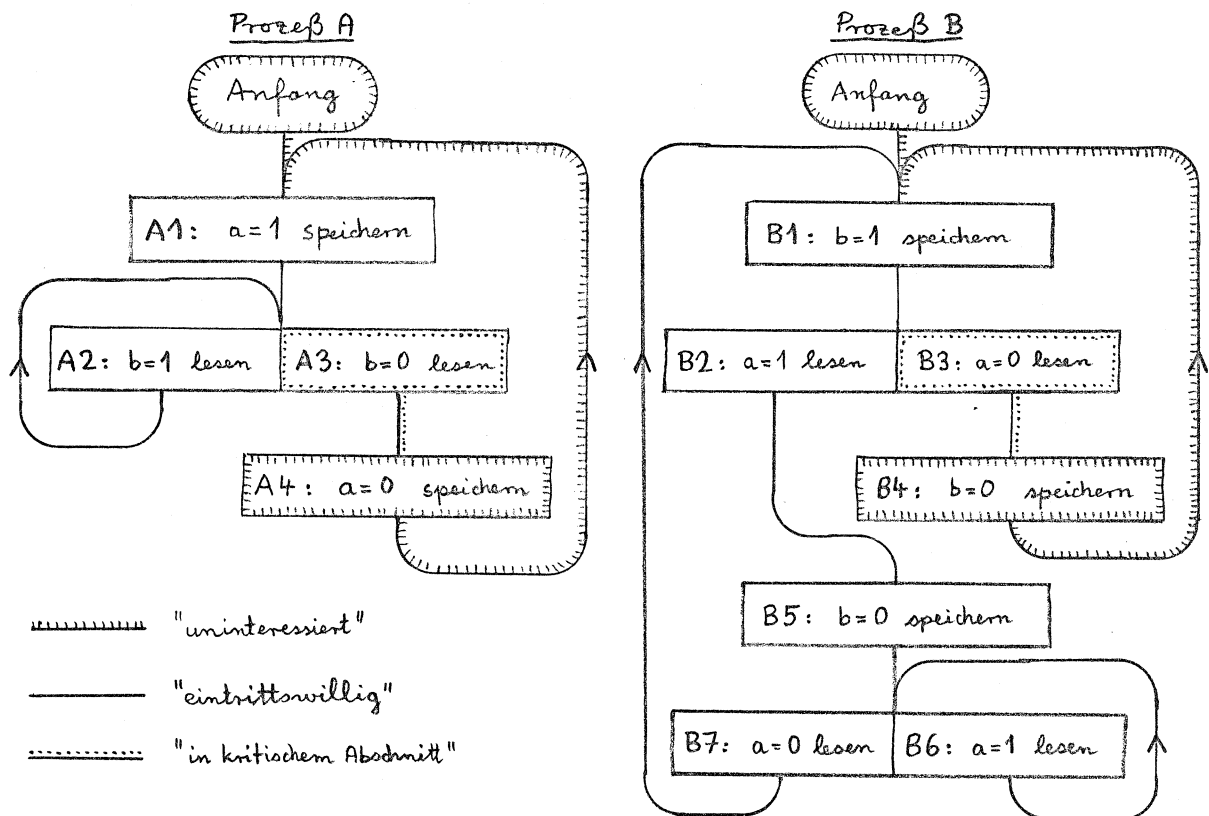


Abb.1. Ablaufpläne für Prozeß A und Prozeß B

dieselbe Zelle führt, sind im Diagramm durch ein Fähnchen in diagonaler Richtung gekennzeichnet.)

Einen Überblick über alle erreichbaren Zustände erhält man auf folgendem Wege: Von jedem als erreichbar erkannten Zustandspunkt aus (am Anfang ist nur Punkt (A4, B4) als erreichbar bekannt) zieht man 2 bzw. 3 Verbindungslinien:

1. Eine für den Fall, daß der nächste Zugriff auf a oder b von Prozeß A herkommt. Der neue Zustand ergibt sich dann aus dem Ablaufplan für Prozeß A und ggf. dem Wert von b, der durch den (konstant bleibenden) Zustand von Prozeß B bestimmt ist.
2. Eine für den Fall, daß der nächste Zugriff auf a oder b von Prozeß B herkommt. Der neue Zustand ergibt sich dann aus dem Ablaufplan für Prozeß B und ggf. dem Wert von a, der durch den (konstant bleibenden) Zustand von Prozeß A bestimmt ist.
3. Wenn es möglich ist, daß der nächste Zugriffszeitpunkt von Prozeß A mit dem nächsten von Prozeß B zusammenfällt, so wird auch für diesen Fall eine Verbindungslinie gezogen. Zum Auffinden des neuen Zustands müssen die Ablaufpläne beider Prozesse herangezogen werden und (nur, falls von beiden Prozessen her ein Lesezugriff beginnt) die Werte von a und b.

Punkte, in die die Verbindungslinien münden, sind auch als erreichbar erkannt. Das Aufsuchen von erreichbaren Zuständen ist beendet, wenn von jedem als erreichbar erkannten Zustand aus die oben erwähnten 2 bzw. 3 Verbindungslinien gezogen sind. Der sich ergebende Zustandsgraph (Abb. 2) läßt erkennen, daß der Punkte(A3, B3) unerreichbar ist; damit ist Behauptung (1*) bewiesen.

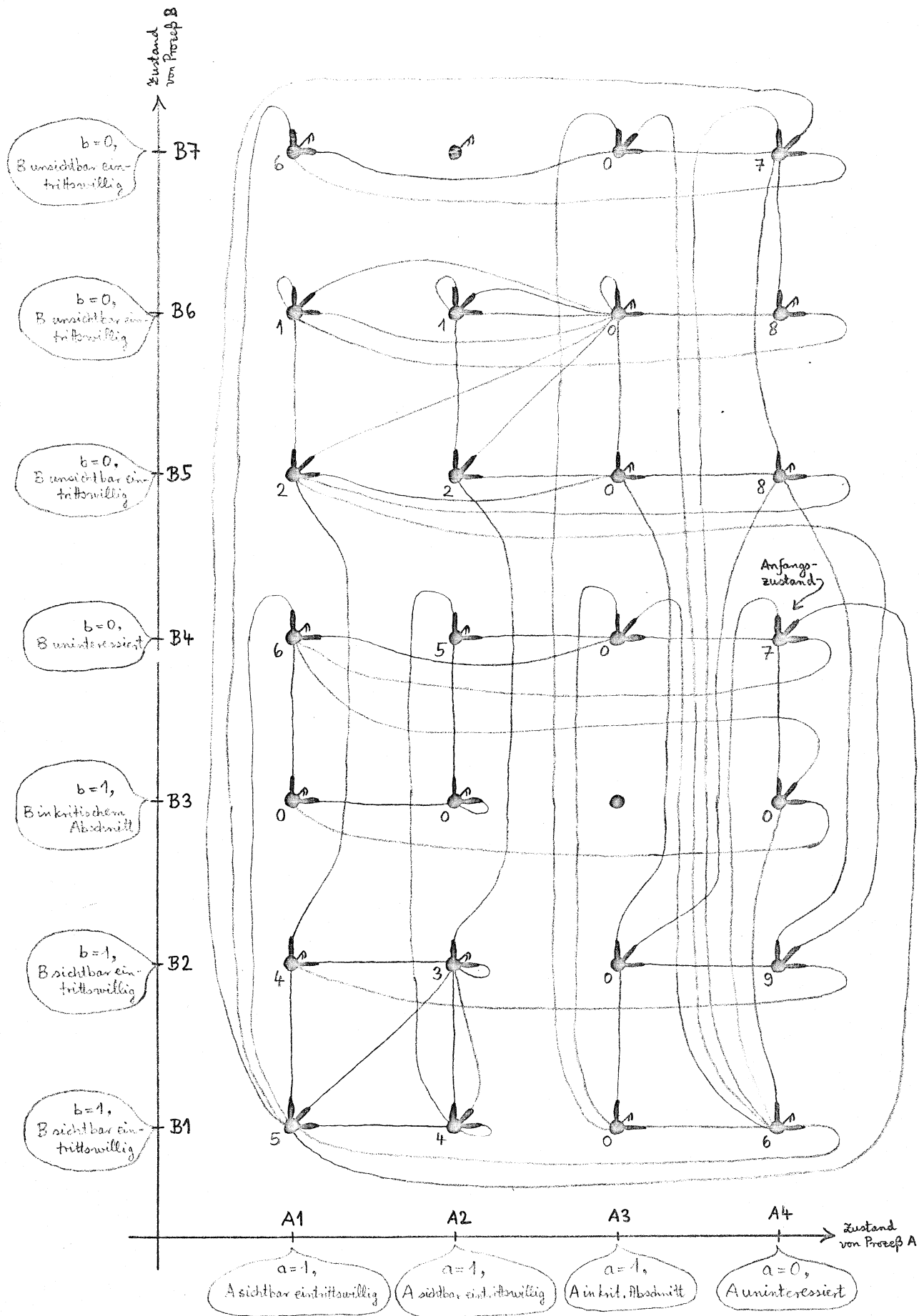


Abb. 2. Zustandsgraph

Im Zustandsgraphen ist bei jedem erreichbaren Zustandspunkt eine Zahl angegeben, die anzeigt, nach wieviel Zustandsänderungen spätestens ein kritischer Zustand (d.h. Zustand mit der Komponente A3 oder B3) erreicht wird.

Die Behauptung (2*) besagt nun, daß man von einem nicht-kritischen Zustand aus, der von (A4, B4) verschieden ist, in absehbarer Zeit in einen kritischen Zustand gelangen muß. Dies sieht man folgendermaßen ein:

1. Die bei den Zustandspunkten vermerkten Zahlen zeigen, daß spätestens nach 9 Zustandsänderungen der nächste kritische Zustand erreicht wird.
2. Die vor dem Eintritt in den nächsten kritischen Zustand durchlaufenen Zustände (im folgenden "Zwischenzustände" genannt) sind unkritische Zustände und von (A4, B4) verschieden, weil (A4, B4) gemäß Zustandsgraph nur von (A3, B4) oder (A4, B3), d.h. von kritischen Zuständen aus, erreicht werden kann.
3. Die Verweildauer in jedem Zwischenzustand ist absehbar, weil z.B. der Zustand (A4, B6) sowie die Zwischenzustände mit der Komponente B1, B2, B5 oder B7 spätestens nach der Ausführung von ein paar Befehlen des Prozesses B verlassen werden und die Zustände (A1, B4), (A1, B6), (A2, B4) und (A2, B6) spätestens nach Ausführung von ein paar Befehlen des Prozesses A. Daß die Ausführung der nächsten paar Befehle bei jedem Prozeß in absehbarer Zeit erfolgt, war für den Beweis der Behauptung (2*) ausdrücklich vorausgesetzt worden.

4. Vergleich mit der DEKKER'schen Methode

a) Zunächst vermutet man, daß die DEKKER'sche Methode durch die Verwendung der zusätzlichen Variablen "turn" folgenden Vorteil hat:

(3*) Ist ein Prozeß in einem kritischen Abschnitt, und der andere Prozeß eintrittswillig, so erfolgt der nächste Eintritt in einen kritischen Abschnitt garantiert vom anderen Prozeß aus.

Dies ist jedoch keineswegs der Fall. Auch bei DEKKER kann ein Prozeß gleich nach Verlassen eines kritischen Abschnittes wieder in einen neuen kritischen Abschnitt eintreten, obwohl der andere Prozeß die ganze Zeit über eintrittswillig ist. (Beispiel: Die zwei Prozesse arbeiten im Zeitmultiplex auf einem Rechnerkern, und der eine Prozeß verläßt zwar den kritischen Abschnitt, tritt aber innerhalb derselben Zeitscheibe wieder in einen neuen kritischen Abschnitt ein, so daß der andere Prozeß von seiner vorübergehenden Eintrittschance gar nichts merkt.)

Solange kritische Abschnitte nur einen geringen Teil der Prozeßablaufzeiten einnehmen, wird man auf Forderung (3*) verzichten.

Will man Forderung (3*) jedoch erfüllen, so muß man dafür sorgen, daß die Zwischenzeit, während der der eine Prozeß dem anderen eine Eintrittschance läßt, so lang ist, daß der andere Prozeß währenddessen die paar Befehle, die für seinen Eintritt in einen kritischen Abschnitt notwendig sind, garantiert ausführen kann. Dies erreicht man, indem man die Zwischenzeit des einen Prozesses nötigenfalls durch das Ausführen zusätzlicher Befehle verlängert; dabei zeigt es sich, daß man bei der DEKKER'schen Methode i.allg. mit ein paar zusätzlichen Befehlen weniger auskommt als bei der hier beschriebenen Methode.

- b) Betrachtet man einen Zeitpunkt, in dem beide Prozesse "sichtbar eintrittswillig" sind (jedoch beide nicht in einem kritischen Abschnitt), so zeigt sich:
- (1) Bei der hier beschriebenen Methode tritt immer Prozeß A als nächster in einen kritischen Abschnitt ein.
 - (2) Bei der DEKKER'schen Methode ist derjenige Prozeß benachteiligt, der zuletzt in einen kritischen Abschnitt war, und der andere Prozeß tritt als nächster in einen kritischen Abschnitt ein.

Dieser augenfällige Unterschied ist jedoch ziemlich bedeutungslos, da es i. allg. sehr unwahrscheinlich ist, daß sich ein Zeitintervall, in dem ein Prozeß "sichtbar eintrittswillig" ist, mit einem ebensolchen des anderen Prozesses überlappt.