

Bisimulationsspiele

Jörn Laun

10. Juli 2006

Schriftliche Ausarbeitung zum Vortrag im Hauptseminar **Spiele in der Informatik**, welches im Sommersemester 2006 an der Uni Stuttgart von Prof. J. Esparza durchgeführt wurde.

In diesem Vortrag soll anhand von nebenläufigen Prozessen und Bisimulationen eine Anwendung spieltheoretischer Methoden in der Informatik vorgestellt werden.

Inhaltsverzeichnis

| | | |
|----------|---|-----------|
| 1 | Einleitung | 1 |
| 1.1 | Vorwort | 1 |
| 2 | Prozessbeschreibungen, CCS | 1 |
| 2.1 | Nebenläufige Prozesse | 1 |
| 2.2 | CCS | 1 |
| 3 | Äquivalenz von Prozessen | 3 |
| 3.1 | Motivierende Beispiele | 3 |
| 3.2 | Bisimulationsäquivalenz | 3 |
| 3.3 | Spieläquivalenz | 4 |
| 3.3.1 | Das Äquivalenzspiel | 4 |
| 3.3.2 | Strategien im Äquivalenzspiel | 5 |
| 3.3.3 | Beziehung zur Bisimulationsäquivalenz | 6 |
| 3.3.4 | Existenz von Gewinnstrategien | 6 |
| 3.3.5 | Beispiele | 7 |
| 4 | Anhang: Mathematische Grundlagen | 9 |
| 4.1 | Wohlordnungen | 9 |
| 4.2 | Ordinalzahlen | 9 |
| | Literatur | 10 |

1 Einleitung

1.1 Vorwort

In den vorangegangenen Vorträgen des Seminars wurden die Grundlagen der Spieltheorie eingeführt und erläutert. Nun soll gezeigt werden, wie sich solche Methoden auf Probleme der theoretischen Informatik anwenden lassen. Als Beispiel hierzu dient das Problem der Äquivalenz paralleler Prozesse, wie in [Sti98] vorgestellt. Der Vortrag orientiert sich in großen Teilen an diesem Artikel, sowie [Sti01].

2 Prozessbeschreibungen, CCS

2.1 Nebenläufige Prozesse

Zur Beschreibung nebenläufiger Prozesse wird der Calculus of Communicating Systems (CCS) verwendet, ein Kalkül, welcher in den siebziger Jahren von Hoare und Milner entworfen wurde. Eine ausführliche Beschreibung verschiedener CCS-Varianten findet sich zum Beispiel in [Win93].

2.2 CCS

Ausgangspunkt ist eine Menge \mathcal{A}' von Aktionen. Diese modellieren Zustandsübergänge von Prozessen. Aktionen dienen aber auch zur Kommunikation. Hierfür wird zusätzlich zu jedem $\kappa \in \mathcal{A}'$ ein $\bar{\kappa}$ eingeführt. Kommunikation findet statt, indem ein Prozess einen κ - und ein anderer parallel einen $\bar{\kappa}$ -Übergang durchführt. Von übertragenen Daten bei einer Kommunikation wird vollständig abstrahiert. Ebenso wird eine Unterscheidung in Sender und Empfänger als unerheblich angesehen und $\bar{\bar{\kappa}} = \kappa$ gesetzt. Eine erfolgte (und damit nach außen nicht sichtbare) Kommunikation wird durch die ausgezeichnete Aktion τ dargestellt. Die entgültige Menge von Aktionen ist also

$$\mathcal{A} := \mathcal{A}' \cup \{\bar{\kappa} : \kappa \in \mathcal{A}'\} \cup \{\tau\}.$$

Definition 2.1 Die Menge \mathcal{P} aller Prozesse ist gegeben als kleinster Fixpunkt der Gleichung

$$\begin{aligned} \mathcal{P} = & \{\kappa.P : \kappa \in \mathcal{A}, P \in \mathcal{P}\} \cup \left\{ \sum_{i \in I} P_i : P_i \in \mathcal{P} \right\} \cup \{(P_1 | P_2) : P_1, P_2 \in \mathcal{P}\} \\ & \cup \{P \setminus K : K \subset \mathcal{A} \setminus \{\tau\}\} \cup \{p : p \stackrel{def}{=} P, P \in \mathcal{P}\} \end{aligned}$$

Die Semantik dieser Prozesse wird mittels der Übergangsrelation $\cdot \xrightarrow{\cdot} \cdot \subset \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ definiert, welche durch folgende Regeln gegeben ist:

| | | |
|------------------|---|---|
| Aktion ausführen | $\frac{}{\lambda.P \xrightarrow{\lambda} P}$ | $P \in \mathcal{P}, \lambda \in \mathcal{A}$ |
| Alternative | $\frac{\exists_{i \in I} P_i \xrightarrow{\lambda} Q}{\sum_{i \in I} P_i \xrightarrow{\lambda} Q}$ | $P_i, Q \in \mathcal{P}, \lambda \in \mathcal{A}$ |
| Parallelität | $\frac{P_1 \xrightarrow{\lambda} P'_1, P_2 \xrightarrow{\lambda} P'_2}{(P_1 P_2) \xrightarrow{\lambda} (P'_1 P_2), (P_1 P_2) \xrightarrow{\lambda} (P_1 P'_2)}$ | $P_1, P_2, P'_1, P'_2 \in \mathcal{P}, \lambda \in \mathcal{A}$ |
| Kommunikation | $\frac{P_1 \xrightarrow{\kappa} P'_1, P_2 \xrightarrow{\bar{\kappa}} P'_2}{(P_1 P_2) \xrightarrow{\tau} (P'_1 P'_2)}$ | $P_1, P_2, P'_1, P'_2 \in \mathcal{P}, \kappa \in \mathcal{A} \setminus \{\tau\}$ |
| Ausschluss | $\frac{P \xrightarrow{\lambda} Q, \lambda, \bar{\lambda} \notin K}{P \setminus K \xrightarrow{\lambda} Q \setminus K}$ | $P, Q \in \mathcal{P}, \lambda \in \mathcal{A}, K \subset \mathcal{A} \setminus \{\tau\}$ |
| Rekursion | $\frac{P \xrightarrow{\lambda} Q, p \stackrel{def}{=} P}{p \xrightarrow{\lambda} Q}$ | $P, Q \in \mathcal{P}, \lambda \in \mathcal{A}$ |

Zur Verkürzung der Schreibweise werden Summen über endlich viele Prozesse auch mit $+$ -Zeichen geschrieben. Der Prozess \sum_{\emptyset} , welcher keinen Übergang machen kann, wird auch mit nil bezeichnet.

Zur Veranschaulichung werden nun verschiedene Beispiele von Prozessen diskutiert. Einige davon sind [Sti98] entnommen.

- Ein Prozess, der sequentiell die zwei Aktionen α und β ausführt:

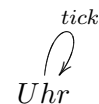
$$\alpha.\beta.\text{nil}$$

Übergangsgraph:

$$\alpha.\beta.\text{nil} \xrightarrow{\alpha} \beta.\text{nil} \xrightarrow{\beta} \text{nil}$$

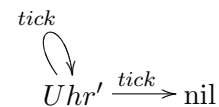
- Endlosschleife am Beispiel einer (sehr einfachen) Uhr:

$$\text{Uhr} \stackrel{def}{=} \text{tick}.\text{Uhr}$$



- Entscheidung: Diese Uhr kann irgendwann stehenbleiben:

$$\text{Uhr}' \stackrel{def}{=} \text{tick}.\text{Uhr}' + \text{tick}.\text{nil}$$



Nun noch zwei Beispiele zu Parallelität und Kommunikation.

- Client/Server: Clients schicken Anfragen (α) an den Server, dieser verarbeitet sie (ω) und sendet eine Antwort (γ).

$$\begin{aligned} S &\stackrel{def}{=} \bar{\alpha}.\omega.\gamma.S \\ C &\stackrel{def}{=} \alpha.\bar{\gamma}.\text{nil} \\ P &\stackrel{def}{=} (S|(C|\dots)) \setminus \{\alpha, \gamma\} \end{aligned}$$

- Kritischer Abschnitt: Arbeitsprozesse C haben einen unkritischen Abschnitt (η), betreten den kritischen Abschnitt (ϵ), fñhrend dort Aktionen aus (κ), verlassen den kritischen Abschnitt wieder (δ) und beginnen dann von vorn. Ein Wächterprozess G kontrolliert, dass keine zwei Prozesse gleichzeitig in den kritischen Abschnitt geraten können.

$$\begin{aligned} W &\stackrel{def}{=} \eta.\epsilon.\kappa.\delta.W \\ G &\stackrel{def}{=} \bar{\epsilon}.\bar{\delta}.G \\ P &\stackrel{def}{=} (G|(W|\dots)) \setminus \{\epsilon, \delta\} \end{aligned}$$

Der Ausschluss von ϵ und δ erzwingt eine interne Synchronisation zwischen Arbeits- und Wächterprozess. Diese Nachrichten können also nicht wie normale Aktionen (wie etwa η oder κ) ausgeführt werden.

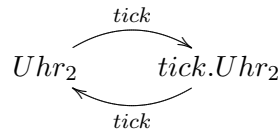
3 Äquivalenz von Prozessen

3.1 Motivierende Beispiele

- Eine weitere Uhr ist gegeben durch

$$Uhr_2 \stackrel{def}{=} tick.tick.Uhr_2$$

Ihr Zustandübergangsgraph



unterscheidet sich deutlich von dem von Uhr . Ein Beobachter, welcher nur das Verhalten der Prozesse (also die geschalteten Transitionen) sieht, könnte Uhr und Uhr_2 dennoch nicht unterscheiden.

- Gegeben seien zwei unterschiedliche Prozesse P, Q . Dann können die Prozesse $(P|Q)$ und $(Q|P)$ zwar genau dieselben Übergänge machen, sind aber (syntaktisch) verschieden.

3.2 Bisimulationsäquivalenz

Es stellt sich also auf ganz natürliche Weise die Frage nach der Äquivalenz von Prozessen. Eine naheliegende Herangehensweise ist zwei Prozesse als äquivalent anzusehen, falls sie sich gegenseitig simulieren können. Formalisiert man diese Idee, so erhält man:

Definition 3.1 Eine binäre Relation $\mathcal{B} \subset \mathcal{P} \times \mathcal{P}$ heißt (starke) **Bisimulation**, falls

- es für alle $(P, Q) \in \mathcal{B}$ und alle Übergänge $P \xrightarrow{\kappa} P'$ ein $Q' \in \mathcal{P}$ gibt, so dass

$$Q \xrightarrow{\kappa} Q' \quad \text{und} \quad (P', Q') \in \mathcal{B}$$

und

-
- es für alle $(P, Q) \in \mathcal{B}$ und alle Übergänge $Q \xrightarrow{\kappa} Q'$ ein $P' \in \mathcal{P}$ gibt, so dass

$$P \xrightarrow{\kappa} P' \quad \text{und} \quad (P', Q') \in \mathcal{B}.$$

Definition 3.2 Zwei Prozesse P und Q heißen **bisimulationsäquivalent**, falls es eine Bisimulation \mathcal{B} gibt, mit $(P, Q) \in \mathcal{B}$.

Man sieht leicht, dass dies eine Äquivalenzrelation ist. Der Beweis wird hier jedoch ausgelassen, da die Aussage im nächsten Kapitel unmittelbar aus den Eigenschaften der Spieläquivalenz folgen wird.

Die oben vorgestellten Uhren sind bisimulationsäquivalent mittels der Bisimulation

$$\mathcal{B} = \{(Uhr, Uhr_2), (Uhr, tick.Uhr_2)\}.$$

3.3 Spieläquivalenz

In diesem Kapitel wird eine Alternative zur Bisimulationsäquivalenz eingeführt.

3.3.1 Das Äquivalenzspiel

Das Äquivalenzspiel wird von zwei Spielern gespielt. Spieler R („Refuter“) versucht zu zeigen, dass zwei gegebene Prozesse P, Q inäquivalent sind, während Spielerin V („Verifier“) deren Äquivalenz zeigen möchte. Das Spiel wird in Runden gespielt, mit jeder Runde ist ein Spielstand (P_i, Q_i) bestehend aus zwei Prozessen P_i und Q_i assoziiert, wobei $P_0 := P$ und $Q_0 := Q$ gesetzt wird. Eine Runde läuft wie folgt ab:

- Zunächst wählt Spieler R eine Transition $P_{i-1} \xrightarrow{\lambda} P_i$. Dann wählt Spielerin V (mit Kenntnis der Wahl von R) eine Transition $Q_{i-1} \xrightarrow{\lambda} Q_i$ (mit demselben λ !).
- ODER -
- Spieler R wählt eine Transition $Q_{i-1} \xrightarrow{\lambda} Q_i$, danach wählt Spielerin V eine Transition $P_{i-1} \xrightarrow{\lambda} P_i$.

Zu jedem Zeitpunkt sind beiden Spielern alle bisherigen (eigenen und gegnerischen) Züge bekannt, das Äquivalenzspiel ist also von vollständiger Information.

Definition 3.3 Ein Spielstand (P, Q) heißt **R-Gewinnstellung**, falls es eine Aktion $\lambda \in \mathcal{A}$ gibt mit

$$\exists P' \in \mathcal{P} P \xrightarrow{\lambda} P' \quad \text{und} \quad \nexists Q' \in \mathcal{P} Q \xrightarrow{\lambda} Q'$$

R gewinnt das Spiel, falls eine R-Gewinnstellung erreicht wird, wenn also R es schafft, eine Transition zu wählen, auf die V nicht reagieren kann. Tritt dies nie ein oder wird ein Spielstand erreicht, in dem R keinen Übergang machen kann, so gewinnt V.

3.3.2 Strategien im Äquivalenzspiel

Äquivalenzspiele zu einem Paar P, Q von Prozessen können durchaus unterschiedliche Ausgänge haben. Beispielsweise kommen für die Prozesse aus dem Uhrenbeispiel folgende beiden Spielverläufe in Frage:

- $(Uhr, Uhr') \xrightarrow{tick} (Uhr, Uhr) \xrightarrow{tick} (Uhr, Uhr) \xrightarrow{tick} \dots$
- $(Uhr, Uhr') \xrightarrow{tick} (Uhr, Uhr') \xrightarrow{tick} (Uhr, nil)$

Im ersten Fall hätte Spielerin V, im zweiten Spieler R gewonnen. Um dennoch einen wohldefinierten Äquivalenzbegriff zu erhalten, betrachtet man Strategien. Beim Äquivalenzspiel haben diese folgende Gestalt:

Definition 3.4

- Eine **Strategie** für Spieler R ist eine Menge von Regeln der Form „Falls der bisherige Spielverlauf $(P_0, Q_0), \dots, (P_{i-1}, Q_{i-1})$ war, so wähle die Transition $X_{i-1} \xrightarrow{\lambda} X_i$.“ (wobei $X \in \{P, Q\}$)
- Eine Strategie für Spielerin V ist eine Menge von Regeln der Form „Falls der bisherige Spielverlauf $(P_0, Q_0), \dots, (P_{i-1}, Q_{i-1})$ war und Spieler R $X_{i-1} \xrightarrow{\lambda} X_i$ gezogen hat, so wähle die Transition $Y_{i-1} \xrightarrow{\lambda} Y_i$.“ (wobei $Y \in \{P, Q\} \setminus \{X\}$)

Definition 3.5 Eine Strategie heißt **Gewinnstrategie** für einen Spieler, falls dieser - solange er die Transitionen wählt, welche die Regelmenge ihm vorschreibt - das Spiel gewinnt, unabhängig davon, welche Züge der Gegenspieler macht.

Definition 3.6 Ein Paar P, Q von Prozessen heißt **spieläquivalent**, falls es eine Gewinnstrategie für Spielerin V gibt.

Im obigen Beispiel (Uhr, Uhr') hat Spieler R eine Gewinnstrategie (ziehe $Uhr' \xrightarrow{tick} nil$). Für das Spiel (Uhr, Uhr_2) hat dagegen Spielerin V eine (ziehe irgendwie).

Lemma 3.7 Spieläquivalenz ist eine Äquivalenzrelation.

Beweis:

- **Reflexivität:** V gewinnt das Spiel (P, P) , indem sie stets R's Züge imitiert.
- **Symmetrie:** V gewinnt das Spiel (Q, P) , indem sie in der Strategie zu (P, Q) die Rollen von P und Q vertauscht.
- **Transitivität:** Hat V Strategien für (P, Q) und (Q, R) , so kann sie immer zunächst einen Zug bei Q durchführen und dann mit Hilfe der zweiten Strategie einen Zug bei P bzw. R ermitteln.

□

3.3.3 Beziehung zur Bisimulationsäquivalenz

Es scheint naheliegend, dass Spiel- und Bisimulationsäquivalenz dasselbe sind. Dass dies richtig ist, zeigt der

Satz 3.8 P und Q sind genau dann spieläquivalent, wenn sie bisimulationsäquivalent sind.

Beweis: „ \Rightarrow “: Man zeigt, dass

$$\mathcal{B} := \{(P, Q) : P \text{ und } Q \text{ sind spieläquivalent}\}$$

eine Bisimulation ist: Ist $(P, Q) \in \mathcal{B}$ und $P \xrightarrow{\lambda} P'$, so kann die entsprechende Regel aus V 's Gewinnstrategie angewandt werden und es folgt, dass $Q \xrightarrow{\lambda} Q'$. Auch für das Paar (P', Q') hat V eine Gewinnstrategie (nämlich dieselbe wie für (P, Q)), daher ist $(P', Q') \in \mathcal{B}$. Entsprechend folgt die zweite Eigenschaft einer Bisimulation.

„ \Leftarrow “: Es existiert eine Bisimulation \mathcal{B} mit $(P, Q) \in \mathcal{B}$. Dann sind die Regeln der Form „Wähle in der i -ten Runde P_i bzw. Q_i so, dass $(P_i, Q_i) \in \mathcal{B}$ liegt.“ eine Gewinnstrategie für V . \square

Korollar 3.9 Bisimulationsäquivalenz ist eine Äquivalenzrelation.

3.3.4 Existenz von Gewinnstrategien

Bisher wurde gezeigt, dass die Äquivalenz von Prozessen gleichbedeutend ist mit der Existenz einer Gewinnstrategie von V für das zugehörige Äquivalenzspiel. Über die Existenz solcher Strategien wurde bislang nichts ausgesagt. Wäre sie jedoch nicht gegeben, so wäre der Begriff der Spieläquivalenz weitgehend nutzlos. Glücklicherweise sichert aber der folgende Satz die Existenz von Gewinnstrategien zu. Man erhält sogar die Existenz gedächtnisloser Gewinnstrategien, so dass die Regeln aus Definition 3.4 eine weitaus einfachere Form bekommen:

Definition 3.10

- Eine **gedächtnislose Strategie** für Spieler R ist eine Menge von Regeln der Form „Falls der aktuelle Spielstand (P, Q) ist, dann wähle die Transition $X \xrightarrow{\lambda} X'$.“ (wobei $X \in \{P, Q\}$)
- Eine gedächtnislose Strategie für Spielerin V ist eine Menge von Regeln der Form „Falls der aktuelle Spielstand (P, Q) ist und Spieler R $X \xrightarrow{\lambda} X'$ gewählt hat, so wähle die Transition $Y \xrightarrow{\lambda} Y'$.“ (wobei $Y \in \{P, Q\} \setminus \{X\}$)

Satz 3.11 Beim Äquivalenzspiel hat einer der Spieler eine gedächtnislose Gewinnstrategie.

Beweis: Betrachte die Menge E aller aus (P, Q) erreichbaren Spielstände:

$$E := \{(P', Q') : P \xrightarrow{\lambda_1} \dots \xrightarrow{\lambda_n} P', Q \xrightarrow{\lambda_1} \dots \xrightarrow{\lambda_n} Q'\}$$

und definiere Teilmengen F_i und F von E , wobei der Index i eine Ordinalzahl ist.

$$\begin{aligned}
 F_0 &:= \{(P, Q) \in E : (P, Q) \text{ ist R-Gewinnstellung}\} \\
 F_{i+1} &:= F_i \cup \bigcup_{\lambda \in \mathcal{A}} \{(P, Q) \in E : \exists P' \in \mathcal{P} P \xrightarrow{\lambda} P' \wedge \forall Q' \in \mathcal{P} Q \xrightarrow{\lambda} Q' \Rightarrow (P', Q') \in F_i\} \\
 &\quad \cup \bigcup_{\lambda \in \mathcal{A}} \{(P, Q) \in E : \exists Q' \in \mathcal{P} Q \xrightarrow{\lambda} Q' \wedge \forall P' \in \mathcal{P} P \xrightarrow{\lambda} P' \Rightarrow (P', Q') \in F_i\} \\
 &\quad \text{für alle Nachfolgerzahlen } i \\
 F_\lambda &:= \bigcup_{i < \lambda} F_i \quad \text{für alle Grenzzahlen } \lambda \\
 F &:= \bigcup_{i \text{ Ordinalzahl}} F_i
 \end{aligned}$$

Außerdem sei für $(P, Q) \in F$

$$rg(P, Q) := \min\{i : (P, Q) \in F_i\}$$

Man beachte, dass dieses Minimum stets existiert!

Nun konstruiert man für ein Äquivalenzspiel (P, Q) eine gedächtnislose Gewinnstrategie:

- Ist $(P, Q) \in F$, so lautet eine Strategie für Spieler R: „Bei Spielstand (P, Q) wähle eine Transition, so dass unabhängig von der Wahl von Spielerin V für den neuen Spielstand (P', Q')

$$rg(P', Q') < rg(P, Q)$$

gilt.“ Nach Konstruktion von F ist dies immer möglich. Nach endlich vielen Schritten wird eine R-Gewinnstellung erreicht.

- Ist $(P, Q) \in E \setminus F$, so kann Spielerin V durch Wahl einer geeigneten Transition stets vermeiden, dass der nachfolgende Spielstand (P', Q') in F liegt. Auch dies ist wegen der Konstruktion von F immer möglich: Angenommen, es gäbe $P \xrightarrow{\lambda} P'$, aber für alle $Q \xrightarrow{\lambda} Q'$ wäre $(P', Q') \in F$. Dann gäbe es also zu jedem solchen Q' ein $i(Q') \in Ord$ mit $(P', Q') \in F_{i(Q')}$. Setzt man $j := \bigcup_{Q'} i(Q')$, so ist $F_j \supset F_{i(Q')}$ für alle Q' , also $(P, Q) \in F_{j+1} \subset F$.

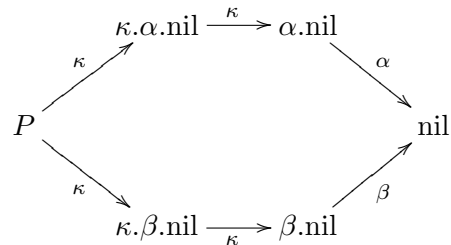
□

3.3.5 Beispiele

Korollar 3.12 Ist in der Situation von Satz 3.11 die Menge F endlich oder $(P, Q) \in F_i$ für ein $i < \omega$, so liefert der Beweis ein konstruktives Verfahren zur Berechnung einer Gewinnstrategie.

Ein sehr einfaches Beispiel hierzu liefert der Prozess

$$P \stackrel{def}{=} \kappa.\kappa.\alpha.nil + \kappa.\kappa.\beta.nil$$



Im Äquivalenzspiel zu (P, P) sind die erreichbaren Spielstände

$$E = \{ (P, P), (\kappa.\alpha.\text{nil}, \kappa.\alpha.\text{nil}), (\kappa.\alpha.\text{nil}.\kappa.\beta.\text{nil}), (\kappa.\beta.\text{nil}, \kappa.\alpha.\text{nil}), (\kappa.\beta.\text{nil}, \kappa.\beta.\text{nil}), (\alpha.\text{nil}, \alpha.\text{nil}), (\alpha.\text{nil}, \beta.\text{nil}), (\beta.\text{nil}, \alpha.\text{nil}), (\beta.\text{nil}, \beta.\text{nil}), (\text{nil}, \text{nil}) \}$$

und

$$\begin{aligned} F_0 &= \{(\alpha.\text{nil}, \beta.\text{nil}), (\beta.\text{nil}, \alpha.\text{nil})\} \\ F_1 &= F_0 \cup \{(\kappa.\alpha.\text{nil}, \kappa.\beta.\text{nil}), (\kappa.\beta.\text{nil}, \kappa.\alpha.\text{nil})\} \end{aligned}$$

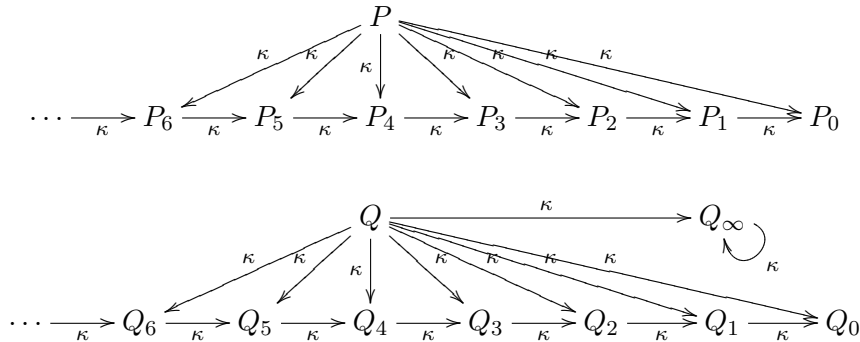
Weitere F_i brauchen wegen $F = F_1$ nicht berechnet werden. Nun ist klar, wie hieraus eine Strategie für V konstruiert werden kann.

Diese Vorgehensweise führt jedoch nicht immer in endlicher Zeit zu einem Ergebnis. Es kann auch vorkommen, dass unendlich viele (auch mehr als ω viele!) F_i benötigt werden¹. Ein Beispiel, bei dem $\omega + 1$ Mengen auftreten, ist das wie folgt definierte Prozesspaar P, Q :

$$\begin{aligned} P_0 &\stackrel{def}{=} \text{nil} \\ P_i &\stackrel{def}{=} \kappa.P_{i-1} \quad (i \in \mathbb{N}) \\ P &\stackrel{def}{=} \sum_{i \in \mathbb{N} \cup \{0\}} P_i \end{aligned}$$

$$\begin{aligned} Q_0 &\stackrel{def}{=} \text{nil} \\ Q_i &\stackrel{def}{=} \kappa.Q_{i-1} \quad (i \in \mathbb{N}) \\ Q_\infty &\stackrel{def}{=} \kappa.Q_\infty \\ Q &\stackrel{def}{=} Q_\infty + \sum_{i \in \mathbb{N} \cup \{0\}} Q_i \end{aligned}$$

Die Übergangsrelationen sind dann:



Die erreichbaren Spielstände des Äquivalenzspiels (P, Q) sind

$$E = \{(P, Q), (P_i, Q_j), (P_i, Q_\infty) : i, j \in \mathbb{N} \cup \{0\}\}.$$

¹Tatsächlich war auch nicht zu erwarten, dass das Äquivalenzproblem einfach würde, da CCS-Prozesse Turing-mächtig sind, siehe [Tau89].

Berechnen der F_i :

$$\begin{aligned} F_0 &= \{(P_0, Q_\infty), (P_0, Q_j), (P_j, Q_0) : j \in \mathbb{N}\} \\ F_1 &= F_0 \cup \{(P_1, Q_\infty), (P_1, Q_j), (P_j, Q_1) : j > 1\} \\ &\vdots \\ F_i &= F_{i-1} \cup \{(P_i, Q_\infty), (P_i, Q_j), (P_j, Q_i) : j > i\} \\ &\vdots \\ F_\omega &= \{(P_i, Q_\infty), (P_i, Q_j) : j \neq i\} \\ F_{\omega+1} &= F_\omega \cup \{(P, Q)\} \end{aligned}$$

Eine Gewinnstrategie für R lautet dann: „Bei (P, Q) ziehe $Q \xrightarrow{\kappa} Q_\infty$, ansonsten ziehe irgendwie.“

4 Anhang: Mathematische Grundlagen

In diesem Kapitel werden in knapper Form die Grundlagen aufgeführt, welche dem Beweis zu Satz 3.11 zugrunde liegen. Ausführlichere Darstellungen sowie detaillierte Beweise finden sich in praktisch jedem Buch über Mengenlehre, beispielsweise [Jec78]. Hinreichend viele Informationen (jedoch ohne Beweise) bietet auch [Wik].

4.1 Wohlordnungen

Definition 4.1 Eine totale Ordnung $<$ einer Menge M heißt Wohlordnung, falls jede nichtleere Teilmenge von M ein kleinstes Element hat.

Lemma 4.2 In einer Wohlordnung $(M, <)$ gibt es keine unendlichen absteigenden Ketten.

4.2 Ordinalzahlen

Definition 4.3 Eine Menge M heißt **Ordinalzahl**, falls sie transitiv ist, d.h. $\forall x \in M x \subset M$ und wohlgeordnet durch \in . Die Klasse aller Ordinalzahlen wird mit Ord bezeichnet.

Definiert man die natürlichen Zahlen als $0 := \emptyset, 1 := \{0\}, 2 := \{0, 1\}, \dots, n := n \cup \{n\}$, so sieht man, dass dies alles Ordinalzahlen sind. Ordinalzahlen können demnach als Verallgemeinerung der natürlichen Zahlen aufgefasst werden.

Jeder Ordinalzahl M kann auf auf kanonische Weise ein Nachfolger $M + 1 := M \cup \{M\}$ zugewiesen werden.

Definition 4.4 Eine Ordinalzahl, welche Nachfolger einer anderen ist, heißt **Nachfolgerzahl**, alle anderen (bis auf die Null) heißen **Grenzzahlen**.

Lemma 4.5

- $M < N \Leftrightarrow M \in N$ definiert eine Wohlordnung auf Ord .
- Für alle $M \in Ord$ gilt: $M = \{N \in Ord : N \in M\} = \{N \in Ord : N < M\}$
- Ist $C \subset Ord$ eine Menge von Ordinalzahlen, so ist $\bigcup C$ eine Ordinalzahl (sogar: $\bigcup C = \sup C$).

Ein Beispiel einer Grenzzahl ist $\omega := \bigcup_{n \in \mathbb{N}} \{n\}$. Es gilt also:

$$1 < 2 < \dots < \omega < \omega + 1 < \omega + 2 := \omega + 1 + 1 < \dots$$

Literatur

- [Jec78] Thomas Jech. *Set theory*. Academic Press, New York, 1978.
- [Sti98] Colin Stirling. Bisimulation, model checking and other games, January 16 1998.
- [Sti01] Colin Stirling. *Modal and Temporal Properties of Processes*. Texts in Computer Science. Springer, 2001.
- [Tau89] Dirk Taubner. *Finite representations of CCS and TCSP programs by automata and Petri nets*. Springer-Verlag New York, Inc., New York, NY, USA, 1989.
- [Wik] Wikipedia-Eintrag zum Thema Ordinalzahlen: <http://de.wikipedia.org/wiki/Ordinalzahl>, zuletzt aufgerufen 01.07.2006.
- [Win93] Glynn Winskel. *The Formal Semantics of Programming Languages*. The MIT Press, Cambridge, Massachusetts, 1993.