Ready trace semantiek voor procesalgebra met prioriteitsoperator

8.2 Ready trace semantiek voor procesalgebra met prioriteitsoperator.

J.C.M. Baeten, *Facultaire Vakgroep Informatica, Universiteit van Amsterdam,*
J.A. Bergstra, *Facultaire Vakgroep Informatica, Universiteit van Amsterdam, Centrale Interfaculteit, Rijksuniversiteit Utrecht,* J.W. Klop, *Centrum voor Wiskunde en Informatica, Amsterdam*

SAMENVATTING: We beschouwen een semantiek voor processen, die ligt tussen bisimulatie semantiek en ready semantiek, genaamd ready trace semantiek. Het voordeel van deze semantiek is dat, terwijl hij nog steeds vrij eenvoudig is, als ready semantiek, we er toch het mechanisme waarbij sommige atomaire acties prioriteit hebben (de θ operator) in kunnen definieren. We laten zien dat in ready semantiek (en a fortiori in failure semantiek) zulk een uitbreiding niet mogelijk is. We beschouwen hier ready trace semantiek in concrete procesalgebra, d.w.z. we laten abstractie en stille stap buiten beschouwing. Ook bekijken we alleen eindige processen. Voor zulke processen geven we een complete axiomatisering van ready trace semantiek.
N.B.: Dit onderzoek werd gedeeltelijk mogelijk gemaakt door steun van ESPRIT project 432, Meteor.

Inleiding.

In Baeten, Bergstra & Klop [1,2] wordt de operator θ ingevoerd in bisimulatiesemantiek. Deze operator introduceert prioriteiten op de atomaire stappen in een proces, en is nuttig voor het specificeren van processen, bv. bij het beschrijven van een interrupt mechanisme. In Bergstra, Klop & Olderog [4] wordt een complete axiomatisering gegeven (voor eindige processen zonder stille stappen) van ready semantiek en failure semantiek.

Het startpunt van dit artikel is nu de vraag, of θ op een consistente manier kan worden toegevoegd aan ready en failure semantiek als in [4]. Dit is niet vanzelfsprekend, aangezien ready en failure semantiek veel meer processen identificeren dan bisimulatiesemantiek. Het blijkt dan ook, dat θ en ready of failure semantiek inconsistent zijn (zie §1).

De volgende vraag is dan, of er een semantiek voor processen is, die "in de buurt zit" van ready en failure semantiek, en waaraan θ consistent kan worden toegevoegd. Er is inderdaad een zeer natuurlijke semantiek met deze eigenschap, nl. RTS, de ready trace semantiek, die ook op zich interessant is. In Pnueli [5] heet RTS "barbed wire semantics" Phillips [6] heeft een soortgelijke semantiek, gebaseerd op failure semantics.

We geven een complete axiomatisering voor eindige processen (zonder τ) voor ready trace semantiek, en geven ook een complete axiomatisering voor de theorie met θ.
Dit artikel kan onafhankelijk gelezen worden, maar het is nuttig tenminste [2] gezien te hebben. Dit artikel is een bewerking van [3], waar ook bewijzen voor enige stellingen gevonden kunnen worden. Voor meer verwijzingen naar de literatuur zie [1,3,4].

1. Ready en failure semantiek zijn inconsistent met prioriteitsoperator.

1.1 Stelling: De axiomastelsels BPA_δ, BPA_δ+R1,2 en BPA_δ+R1,2+S (zie tabel 1) zijn complete axiomatiseringen van respecievelijk bisimulatiesemantiek, ready semantiek en failure semantiek op eindige processen (zonder τ).

BPA_δ	$x + y = y + x$	A1
	$(x + y) + z = x + (y + z)$	A2
	$x + x = x$	A3
	$(x + y)z = xz + yz$	A4
	$(xy)z = x(yz)$	A5
	$x + \delta = x$	A6
	$\delta x = \delta$	A7
BPA_δ + R1,2	$a(bx + u) + a(by + v) = a(bx + by + u) + a(bx + by + v)$	R1
	$a(b + u) + a(by + v) = a(b + by + u) + a(b + by + v)$	R2
BPA_δ+R1,2+S	$ax + a(y + z) = ax + a(x + y) + a(y + z)$	S

Tabel 1.

We hebben een (meestal eindige) verzameling constanten A, het actiealfabet, $A_\delta = A \cup \{\delta\}$ (dus $\delta \notin A$); de binaire operator + is alternatieve compositie, keuze; de binaire operator $\cdot$ is sequentiële compositie; δ is een speciale constante die deadlock aangeeft. a,b zijn elementen van A, en x,y,z,u,v zijn willekeurige processen. We hebben hier in wezen de standaard definities van bisimulatie-, ready en failure semantiek. Wel hebben we twee terminatie-mogelijkheden (een proces kan eindigen in δ, of met een echte actie $a \in A$). Voor meer details verwijzen we naar [4].

1.2 In Baeten, Bergstra & Klop [1] of [2] wordt de **prioriteitsoperator θ**

geintroduceerd, waarmee het mogelijk is uit te drukken dat bepaalde acties prioriteit of voorrang hebben bij een keuze of som. Prioriteiten worden gegeven d.m.v. een partiële ordening $<$ op A_δ, waarbij δ altijd minimaal is. Als we bv. $b<a$ hebben, dan geldt:

$$\theta(ax + by + z) = \theta(ax + z).$$

Ook geldt voor θ bv.

$$\theta(ax) = a\cdot\theta(x), \text{ en } \theta(ax + ay) = a\cdot\theta(x) + a\cdot\theta(y).$$

Verderop zullen we een eindige en complete axiomatisering geven voor θ, maar op dit moment zijn de bovenstaande eigenschappen voldoende. In [1] en [2] werd aangetoond dat θ belangrijk is voor het modelleren van zaken als interrupts. In die artikelen werd θ gedefiniëerd in bisimulatie-semantiek. De vraag is nu, of θ consistent kan worden toegevoegd aan ready en failure semantiek. Met "consistent" bedoelen we, dat we geen gelijkheid kunnen afleiden tussen twee gesloten termen, die een verschillend trace hebben. Een trace is een mogelijke rij acties, die een proces kan uitvoeren. We zullen bewijzen, dat ready en failure semantiek met θ inconsistent is in deze zin.

1.3 Stelling: Zowel ready als failure semantiek met θ is inconsistent. Voldoende om deze inconsistentie af te leiden zijn axioma's BPA_δ + R1 + de gelijkheden voor θ in 1.2.
Bewijs: Zij $s \equiv a(bc + d) + a(be + f)$ en $t \equiv a(be + d) + a(bc + f)$ (voor zekere atomen a,b,c,d,e,f) en stel dat we prioriteiten $f<b<d$ hebben. Met R1 krijgen we $s=t$, maar $\theta(t) = ad + abe$ en $\theta(s) = ad + abc$, zodat $\theta(t)$ en $\theta(s)$ niet alleen een verschillend trace hebben, maar zelfs een verschillend alfabet.

2. Ready trace semantiek: een model voor eindige processen.

We definiëren in deze sectie de **ready trace equivalentie** op procesgrafen.

2.1 Definitie: Zij **H** de verzameling van eindige, acyclische procesgrafen, met de zijden gelabeld met elementen van A, en die in **δ-normaal vorm** zijn, d.w.z. een δ-zijde mag alleen voorkomen naar een eindpunt toe, en mag geen alternatief hebben. We veronderstellen de definitie van $+, \cdot$ op procesgrafen als bekend (zie [4]).

Voor de volledigheid geven we nu eerst de definitie van de ready verzameling $R[\![g]\!]$ van een procesgraaf g:
2.2 Definitie: Zij A^* de verzameling woorden over A (de verzameling eindige traces, zonder δ). Het lege woord geven we aan met λ. Zij $g \in \mathbf{H}$. De

ready verzameling van g, notatie $R[\![g]\!]$, is de kleinste verzameling die aan de volgende eis voldoet:

als $g \xrightarrow{\sigma} h$, dan $(\sigma, I(h)) \in R[\![g]\!]$.

Hier is $\sigma \in A^*$, $g \xrightarrow{\sigma} h$ (h is een **afleiding** van g via σ) als er een pad is, dat het woord σ bepaalt, van de wortel van g naar de wortel van de subgraaf h. Verder is I(h) de verzameling **initiële stappen** van h; als h alleen uit een δ-stap bestaat, geldt $I(h) = \emptyset$, en als h **0** is, de triviale graaf zonder zijden, geldt $I(h) = \{\varepsilon\}$. Hier is ε een formeel symbool dat succesvolle terminatie aanduidt.

2.3 Voorbeeld: Zij g de procesgraaf in fig. 1. Dan geldt
$R[\![g]\!] = \{(\lambda, \{a,b\}), (a, \emptyset), (a, \{c\}), (ac, \{\varepsilon\}), (b, \{\varepsilon\})\}$.

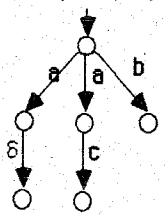

Fig. 1.

2.4 Definitie: Procesgrafen g en h zijn **ready equivalent** als $R[\![g]\!] = R[\![h]\!]$. Notatie: $g \equiv_R h$.

In Bergstra, Klop & Olderog [4] wordt bewezen dat ready equivalentie ook een congruentie is m.b.t. +, ·, en we hebben de isomorfie

$$\mathbf{H}(+,\cdot)/\equiv_R \;\simeq\; I(BPA_\delta + R1,2)$$

(hierbij is I(−) de initiële algebra van −).
Nu zullen we ready trace semantiek (RTS) gaan bespreken; daarbij worden minder processen geïdentificeerd dan in ready semantiek (RS). Het essentiële verschil is, dat terwijl RS gebaseerd is op het begrip van een ready paar (σ,X) (zie fig. 2a, volgende bladzijde), RTS gebaseerd is op het begrip van een **ready trace** (zie fig. 2b, volgende bladzijde), waarbij ook de "tussenliggende" ready verzamelingen langs trace σ gegeven worden.

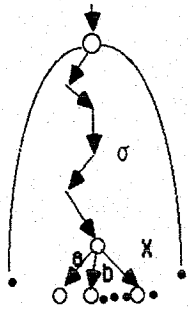

Fig. 2a

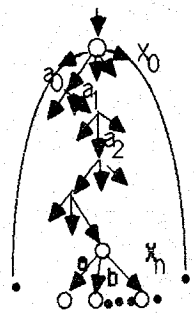

Fig. 2b

2.5 Definitie: (i) Zij g een procesgraaf. Een **pad** π in g, dat begint in de wortel s_0 van g, is een alternerende rij knopen en gelabelde zijden in g:

$$\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots \xrightarrow{a_{n-1}} s_n$$

met $a_i \in A$ (dus $\neq \delta$). Hier is $n \geq 0$; als $n=0$ hebben we het lege pad. Alle paden in dit artikel zullen beginnen in de wortel, dus dat zullen we er niet steeds bijzetten. Een pad is **maximaal** als het niet meer verlengd kan worden, d.w.z. het eindigt in een eindknoop (**0**) of in een knoop waar een δ-stap begint.
(ii) als s een knoop is van g, is $(g)_s$ de **subgraaf** van g met wortel s en met zijden alle zijden die bereikbaar zijn vanuit s.
(iii) I(g) is weer de verzameling initiële stappen van g, met $I(\mathbf{0}) = \{\varepsilon\}$ en $I(\delta) = \emptyset$. In plaats van $I((g)_s)$ schrijven we ook I(s).

2.6 Definitie: (i) Zij g een procesgraaf en $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots \xrightarrow{a_{n-1}} s_n$ een pad in g. Dan is **rt(π)**, de **ready trace** die hoort bij π, de alternerende rij ready verzamelingen $I(s_i)$ en stappen a_i:

$$I(s_0),a_0,I(s_1),a_1,\ldots,a_{n-1},I(s_n)$$

met $n \geq 0$. Soms zullen we de notatie $(\sigma;\vec{X})$ gebruiken voor zo'n ready trace, dan is $\sigma = a_0a_1\ldots a_{n-1}$ en $\vec{X} = I(s_0),I(s_1),\ldots,I(s_n)$. De ready trace die hoort bij het lege pad van g is gewoon I(g), of in de $(\sigma;\vec{X})$ notatie, $(\lambda;I(g))$.
(ii) De **ready trace verzameling** van g, notatie $RT[\![g]\!]$, is

{rt(π) : π een pad in g, beginnend in de wortel}.

(iii) $g \equiv_{RT} h$ als $RT[\![g]\!] = RT[\![h]\!]$, dan noemen we g en h **ready trace equivalent**.

2.7 Voorbeeld: (i) zij g,h als in fig. 3. Dan geldt $g \equiv_{RT} h$.

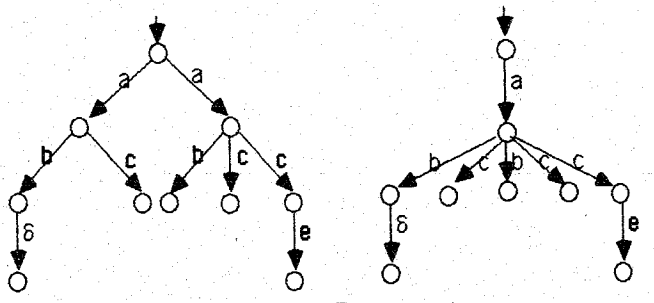

Fig. 3.

(ii) g,h als in fig. 4 zijn *niet* ready trace equivalent (N.B.: dit was het tegenvoorbeeld in stelling 1.3), want $RT[\![g]\!]$ heeft het ready trace $\{a\},a,\{d,b\},b,\{c\},c,\{\varepsilon\}$, dat er niet is in $RT[\![h]\!]$.

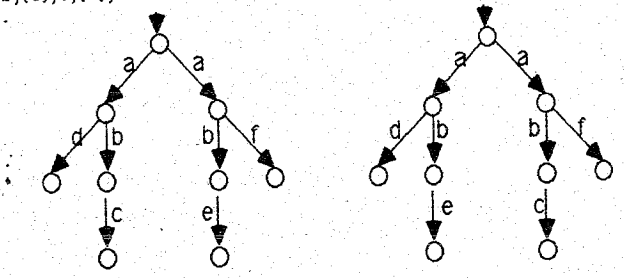

Fig. 4.

2.8 Opmerking: In feite, omdat we werken met eindige acyclische procesgrafen, was het voldoende geweest om in de definitie van $\equiv_{RT}$ alleen ready traces te beschouwen die horen bij *maximale* paden. De definitie die we nu hebben, beschouwt ook "prefixen" van zulke ready traces, en loopt vooruit op het werken met *oneindige* processen - hetgeen we in dit artikel niet zullen doen.

2.9 Opmerking: Een handzame intuïtie voor een ready trace is de volgende: stel u een interactieve sessie met een machine voor. Bij het begin van de sessie geeft de machine een menu van alle mogelijke acties die de gebruiker mag uitvoeren (I(g)); één van deze wordt gekozen (a_0). Vervolgens geeft de machine weer een menu van de opties die dan mogelijk zijn, en zo verder. Op elk moment mag de gebruiker de sessie beëindigen, en laat dan de machine achter met op het scherm het laatste menu ($I(s_n)$).

Een ready trace is het verslag van zo'n sessie.

We zullen verderop de stelling zien dat $\equiv_{RT}$ ook een *congruentie* is op **H** m.b.t. tot de operatoren +,·, maar ook m.b.t. operatoren $\|,\lfloor\!\lfloor,\lfloor,\partial_H$ en θ. De definitie van θ op procesgraaf geven we nu.

2.10 Definitie: Zij een partiële ordening $<$ op A_δ gegeven zodat $\delta<a$ voor elke $a\in A$. De θ die hoort bij deze ordening definiëren we als volgt: $\theta(g)$ is de procesgraaf die we verkrijgen uit de graaf g door
(i) alle zijden uit een knoop s weg te laten met een label 'a' die gemajoreerd wordt door een label 'b' van een andere zijde die uit s vertrekt;
(ii) alle delen van g weg te laten die zodoende onbereikbaar zijn geworden.

2.11 Voorbeeld: Zij $a>b$ en $a>c$. Met g als in fig. 5a, wordt $\theta(g)$ als in fig. 5b.

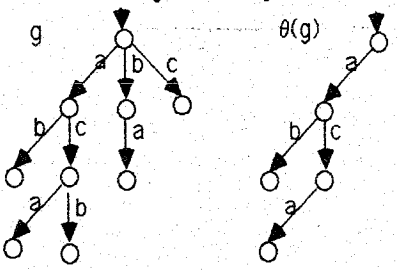

Fig. 5a Fig. 5b

2.12 Stelling: $\equiv_{RT}$ is een congruentie op $\mathbf{H}(+,\cdot,\|,\lfloor\!\lfloor,\lfloor,\partial_H,\theta)$.

Bewijs: Het bewijs gaat met zgn. **procesgraaf transformaties**. T.o.v. een drietal zulke transformaties kunnen we een RTS-normaalvorm definiëren, en vervolgens bewijzen dat equivalentie t.o.v. deze transformaties overeenkomt met ready trace equivalentie. Dan bewijzen we dat elk van de transformaties zich goed gedraagt t.o.v. alle operatoren. Het volledige bewijs is te vinden in [3].

We zullen nog twee hulpoperatoren op **H** definiëren, waarmee we θ kunnen axiomatiseren, en waarmee we een regel eigen aan RTS kunnen formuleren.
2.13 Definitie: Zij g,h procesgrafen. Dan is $g\lhd h$ (g **tenzij** h) het resultaat van het weglaten van alle *initiële* stappen in g, die gemajoreerd worden (in de partiële ordening) door een *initiële* stap in h (en vervolgens weer onbereikbare delen weg te laten).

2.14 Voorbeeld: Zij $a<b<c$. Dan geldt:

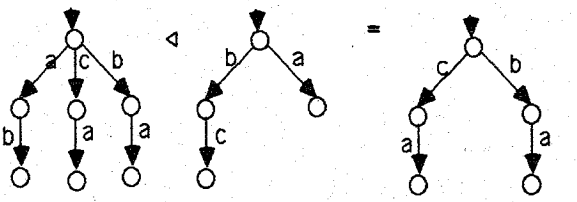

Fig. 6.

2.15 Definitie: π_n is de n-de **projectie-operator** ($n \geq 1$), die alle takken na n stappen afkapt (π_n is gedefinieerd op grafen, door ze eerst uit te rollen tot een boom). Zie fig. 7.

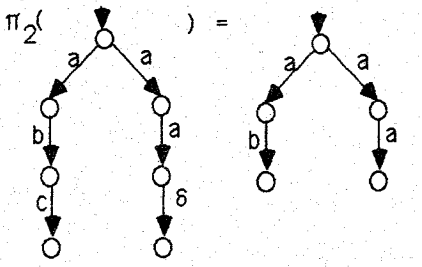

Fig. 7.

Stelling 2.12 geldt ook voor de hulpoperatoren ◁ en π_n.

3. Een complete axiomatisering van ready trace semantiek.

M.b.v. de stellingen in §2 is het nu niet moeilijk, een complete axiomatisering voor ready trace semantiek op eindige processen zonder τ te geven. Dit is het axiomastelstel RTS in tabel 2 (op de volgende bladzijde). Het bovenste deel van tabel 2 is ACP, de Algebra van Communicerende Processen (met $a,b \in A_\delta$). ACP bevat BPA_δ. De π_n zijn de projectie-operatoren; daarvan hebben we alleen π_1 nodig om de ready trace regel RTR te formuleren. Voor eindige processen is RTR equivalent met de regel

$$\frac{\pi(x) = \pi(y)}{a(x + y) = ax + ay}$$

3.1 Lemma: BPA_δ + PR1-4 + RTR is een complete axiomatisering van $\mathbf{H}(+,\cdot,\pi_n)/\equiv_{RT}$.

Bewijs: Met procesgraaftransformaties, zie [3].

$x + y = y + x$	A1
$(x + y) + z = x + (y + z)$	A2
$x + x = x$	A3
$(x + y)z = xz + yz$	A4
$(xy)z = x(yz)$	A5
$x + \delta = x$	A6
$\delta x = \delta$	A7
$a \mid b = b \mid a$	C1
$(a \mid b) \mid c = a \mid (b \mid c)$	C2
$\delta \mid a = \delta$	C3
$x \Vert y = x \mathbin{\lfloor\!\lfloor} y + y \mathbin{\lfloor\!\lfloor} x + x \mid y$	CM1
$a \mathbin{\lfloor\!\lfloor} x = ax$	CM2
$ax \mathbin{\lfloor\!\lfloor} y = a(x \Vert y)$	CM3
$(x + y) \mathbin{\lfloor\!\lfloor} z = x \mathbin{\lfloor\!\lfloor} z + y \mathbin{\lfloor\!\lfloor} z$	CM4
$ax \mid b = (a \mid b)x$	CM5
$a \mid bx = (a \mid b)x$	CM6
$ax \mid by = (a \mid b)(x \Vert y)$	CM7
$(x + y) \mid z = x \mid z + y \mid z$	CM8
$x \mid (y + z) = x \mid y + x \mid z$	CM9
$\partial_H(a) = a$ als $a \notin H$	D1
$\partial_H(a) = \delta$ als $a \in H$	D2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
$\partial_H(xy) = \partial_H(x) \cdot \partial_H(y)$	D4
$\pi_n(a) = a$	PR1
$\pi_1(ax) = a$	PR2
$\pi_{n+1}(ax) = a \cdot \pi_n(x)$	PR3
$\pi_n(x + y) = \pi_n(x) + \pi_n(y)$	PR4
$\dfrac{\pi_1(x) = \pi_1(y)}{z(x + y) = zx + zy}$	RTR

Tabel 2. RTS

De uitbreiding met de prioriteitsoperator θ kan gemakkelijk verwezenlijkt worden aan de hand van de axiomatisering van ACP_θ in [1] of [2]. Het axiomastelsel ACP_θ bestaat uit ACP (bovenste deel van tabel 2) plus de

negen axioma's in tabel 3. Het axiomastelstel bestaande uit RTS plus de axioma's in tabel 3 zullen we RTS_θ noemen.

$a \lhd b = a$	als $\neg(a<b)$	P1
$a \lhd b = \delta$	als $a<b$	P2
$x \lhd yz = x \lhd y$		P3
$x \lhd (y + z) = (x \lhd y) \lhd z$		P4
$xy \lhd z = (x \lhd z)y$		P5
$(x + y) \lhd z = x \lhd z + y \lhd z$		P6
$\theta(a) = a$		TH1
$\theta(xy) = \theta(x) \cdot \theta(y)$		TH2
$\theta(x + y) = \theta(x) \lhd y + \theta(y) \lhd x$		TH3

Tabel 3.

3.2 Stelling: RTS_θ is een complete axiomatisering van het graafmodel $\mathbf{H}(+,\cdot,\|,\lfloor\!\lfloor,\lfloor,a_H,\pi_n,\lhd,\theta)/\equiv_{RT}$. Bewijs: zie [3].

3.3 Gevolg: RTS_θ is consistente axiomatisering (in de zin van 1.2).

4. Een expliciet model voor ready trace semantiek.

We hebben een model voor RTS gedefiniëerd, bestaande uit equivalentieklassen procesgrafen. We kunnen dit model ook op een meer directe wijze presenteren, nl. door ready trace verzamelingen zelf als elementen te nemen (zonder de onderliggende procesgrafen te noemen). Hiervoor moeten we enkele afsluitingseigenschappen van ready trace verzamelingen formuleren (vlg. de procedure in [4] voor failure semantiek). Op die wijze krijgen we een zgn. expliciet model voor RTS, isomorf met het bovenstaande graafmodel. Daarvoor moeten we wel de operatoren $+,\cdot,\|,\lfloor\!\lfloor,a_H,\theta,\lhd,\pi_n$ rechtstreeks op ready trace verzamelingen definiëren. Dit zullen we alleen doen voor θ.

4.1 Definitie: (i) Zij $X_0,a_0,X_1,a_1,\ldots,a_{n-1},X_n$ een alternerende rij van verzamelingen X_i $(i=0,\ldots n)$ en acties a_i $(i=0,\ldots,n-1)$, met $n \geq 0$. Deze rij is een **ready trace** als (1) $a_i \in X_i$ $(i<n)$; (2) $X_0,\ldots,X_{n-1} \subseteq A$; (3) $X_n \subseteq A$ of $X_n = \{\varepsilon\}$ $(\varepsilon \notin A)$. In het volgende gebruiken we weer de $(\sigma;\vec{X})$ notatie van 2.6:
(ii) Zij $\mathcal{X}$ een collectie ready traces. $\mathcal{X}$ is een **ready trace verzameling** als $\mathcal{X}$ voldoet aan de volgende voorwaarden:

(1) als $(\sigma;\vec{X}) \in \mathcal{X}$ en $(\rho;\vec{Y}) \in \mathcal{X}$ dan $X_0 = Y_0$ (**wortelconditie**);

(2) als $(\sigma;\vec{X})$ een ready trace is, en $(\sigma\rho;\vec{X},\vec{Y}) \in \mathcal{X}$, dan $(\sigma;\vec{X}) \in \mathcal{X}$ (**prefix conditie**);

(3) als $(a_0....a_{k-1};X_0,....,X_k) \in \mathcal{X}$ en $a_k \in X_k$ $(a_k \neq \varepsilon)$, dan $(a_0....a_{k-1}a_k;$ $X_0,....,X_k,X_{k+1}) \in \mathcal{X}$ voor zekere X_{k+1} $(\subseteq A$ of $=\{\varepsilon\})$ (**continuatieconditie**).

Het is duidelijk dat voor elke procesgraaf g, $RT[\![g]\!]$ een ready trace verzameling is met deze definitie. Omgekeerd kunnen we met elke ready trace verzameling $\mathcal{X}$ een procesboom $g_{\mathcal{X}}$ associëren. We zullen nu nog de operator θ definiëren op deze ready trace verzamelingen, en verduidelijken daarmee wederom waarom θ compatibel is met RTS - en niet met grovere semantieken als ready semantiek of failure semantiek.

4.2 Definitie: Zij $<$ een partiële ordening op A_δ (met δ minimaal).

(i) Zij $X \subseteq A$. $\theta(X)$ is de verzameling *maximale* (t.o.v. $<$) elementen in X.
Als $X=\{\varepsilon\}$ geldt $\theta(X)=X$. Als $\vec{X}=X_0,....,X_n$, dan $\theta(\vec{X})= \theta(X_0),....,\theta(X_n)$.

(ii) Zij $(\sigma;\vec{X})$ een ready trace. Dan hoeft $(\sigma;\theta(\vec{X}))$ niet een ready trace te zijn, omdat misschien niet aan eigenschap (1) van definitie 5.1 voldaan is $(\sigma;\theta(\vec{X}))$ heeft echter wel een maximaal prefix, dat nog wel een ready trace is. We definiëren dat $\boldsymbol{\theta(\sigma;\vec{X})}$ dit maximaal prefix ready trace is van $(\sigma;\theta(\vec{X}))$. Een voorbeeld: als $a<b$, dan $\theta(\{a,b\},b,\{a,b\},a,\emptyset) = \{b\},b,\{b\}$.

(iii) Zij $\mathcal{X}$ een ready trace verzameling. Dan $\theta(\mathcal{X}) = \{\theta(\sigma;\vec{X}) : (\sigma;\vec{X}) \in \mathcal{X}\}$.

Nu stellen we (zonder bewijs) dat θ en $RT[\![\ ]\!]$ commuteren:

4.3 Claim: Zij g een procesgraaf. Dan geldt $RT[\![\theta(g)]\!] = \theta(RT[\![g]\!])$.

We zien dat het voor deze expliciete definitie van de operator θ op ready trace verzamelingen essentiëel is, dat we de tussenliggende ready verzamelingen in een ready trace beschikbaar hebben.

Conclusie.

In het voorgaande is geconstateerd dat het toevoegen van een zeer natuurlijke operator, de prioriteitsoperator, aan een al even natuurlijke en veel bestudeerde proces semantiek, de failure semantiek, verrassend genoeg een inconsistentie oplevert. Aan een fijnere (i.e. meer processen onderscheidende) proces semantiek, de bisimulatie semantiek, kan de prioriteitsoperator wel consistent toegevoegd worden. Omdat grovere semantieken meer geschikt zijn voor gemakkelijke procesverificaties, leidt dit tot de vraag hoe 'grof' de semantiek gemaakt kan worden terwijl er toch nog compatibiliteit met de prioriteitsoperator is. Een goede keuze

blijkt 'ready trace semantiek' te zijn (Pnueli's 'barbed wire' semantiek). Deze semantiek heeft een bevredigende intuitieve betekenis en kan door een eenvoudige uitbreiding van de axioma's voor bisimulatiesemantiek geaxiomatiseerd worden; het hier gegeven axiomasysteem is voor eindige processen zelfs compleet.

Feiten als boven gerapporteerd passen in het algemene kader van 'vergelijkende concurrency semantiek', een onderneming waarbij gepoogd wordt diverse (algebraische) processemantieken onder één noemer te brengen.

Literatuurverwijzingen.

[1] J.C.M.Baeten, J.A.Bergstra & J.W.Klop, *Syntax and defining equations for an interrupt mechanism in process algebra*, CWI rapport CS-R8503, Amsterdam 1985.
[2] J.C.M.Baeten, J.A.Bergstra& J.W.Klop, *Proces algebra met interrupt mechanisme*, NGI-SION Symposium 1985, blz. 129 - 135.
[3] J.C.M.Baeten, J.A.Bergstra& J.W.Klop, *Ready trace semantics for concrete process algebra with priority operator*, CWI rapport CS-R8517, Amsterdam 1985.
[4] J.A.Bergstra, J.W.Klop & E.-R.Olderog, *Readies and failures in the algebra of communicating processes*, CWI rapport CS-R8523, Amsterdam 1985.
[5] A.Pnueli, *Linear and branching structures in the semantics and logics of reactive systems*, Proc. 12th ICALP, Ed. W.Breuer, Springer LNCS 194, 1985, blz. 15-32
[6] I.C.C.Phillips, *Refusal testing*, Research Report DoC 85/17, Imperial College, University of London, 1985.