stichting
mathematisch
centrum

MC

AFDELING INFORMATICA

IN 10/76 FEBRUARI

L. AMMERAAL

ENIGE HULPMIDDELEN BIJ KORREKTHEIDSBEWIJZEN

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

AMS(MOS) subject classification scheme (1970): 68-00, A40

ACM-Computing Reviews-categories: 4.2, 5.24

Enige hulpmiddelen bij korrektheidsbewijzen

door

L. Ammeraal

ABSTRACT

Om de korrektheid van een programma of van een gedeelte daarvan uit de programmatekst zelf te bewijzen dient men over enig gereedschap in de vorm van bewijsregels te beschikken. Een aantal bekende en minder bekende bewijsregels wordt besproken en door voorbeelden geïllustreerd. Onderscheid wordt gemaakt tussen "voorwaartse" en "achterwaartse" regels. Tenslotte worden enige interessante eigenschappen van bewijsregels genoemd.

KEY WORDS & PHRASES: *Correctness proof, program verification*

1. INLEIDING

De korrektheid van een programma dient bij voorkeur te blijken uit de programmatekst zelf. Door een goede programmeertaal te kiezen en door overzichtelijk oftewel "gestructureerd" te programmeren komt men al een eind in de goede richting. Toch kan het gewenst zijn de korrektheid van een programma of van een onderdeel daarvan op meer expliciete wijze aan te tonen. Onderzoek op dit gebied is verricht door o.a. FLOYD, HOARE, DIJKSTRA, NAUR, WIRTH, MANNA, MILLS en DE BAKKER. In principe bestaat een korrektheidsbewijs uit het formuleren en kontroleren van voorwaarden die aan waarden van variabelen worden gesteld. Een verband tussen zulke voorwaarden vlak vóór en vlak na de uitvoering van een programmastatement noemt men wel een "bewijsregel". We zullen een aantal van zulke bewijsregels formuleren en toelichten. Hierbij zal niet worden geschroomd enkele bruikbare regels te behandelen die men niet dagelijks in de literatuur tegenkomt. Een aantal eigenschappen van bewijsregels zijn te interessant om onvermeld te blijven. Zij geven antwoord op vragen die vanzelf bij de gebruiker van de regels opkomen. Er zal van worden afgezien een onderscheid aan te geven tussen zaken die men overal in de literatuur over deze materie aantreft en mogelijke nieuwigheden. Het is in de informatica soms moeilijk uit te maken of een vondst echt nieuw is, doordat er nog weinig uniformiteit in de terminologie bestaat.

Het is niet de bedoeling van deze bijdrage, spectaculaire bewijzen van realistische programma's te behandelen. De voorbeelden zijn met opzet uiterst eenvoudig gehouden. Zij hebben slechts ten doel aannemelijk te maken dat de behandelde bewijsregels geldig en *in principe* bruikbaar zijn.

Voor voorbeelden die dichter bij de realiteit staan en het praktisch nut van korrektheidsbeschouwingen beter illustreren kan onder meer naar [7] en [8] worden verwezen.

2. VOORWAARTSE EN ACHTERWAARTSE BEWIJSREGELS

De volgende regel is intuïtief meteen duidelijk:

(1) $\{x > 5\}\ \ x := x + 1\ \ \{x > 6\}$.

Hierin is $x := x + 1$ een programmastatement. Als vlak vóór de uitvoering van deze statement $x > 5$ waar is, dan is vlak erna $x > 6$ waar. De algemene gedaante van (1) is

(2) $\{P\}\ \ S\ \ \{Q\}$.

Hierin zijn P en Q voorwaarden, gesteld aan de "toestand" vlak vóór resp. vlak na de uitvoering van statement S. Met "toestand" wordt het verband tussen programmavariabelen en hun waarde bedoeld. Kortheidshalve zullen we spreken over "vóór S" of "vooraf" in plaats van "vlak voor de uitvoering van S". Een analoge betekenis zullen "na S" en "achteraf" hebben. We definiëren de betekenis van (2) als:

(3) Als de toestand vooraf aan P voldoet dan is S voor deze toestand gedefinieerd en dan voldoet de toestand achteraf aan Q.

Volgens deze afspraak geldt ook:

(4) $\{x > 5\}\ \ x := x + 1\ \ \{x > 0\}$.

Het toelaten van "zwakke" uitdrukkingen als (4) is soms ongewenst. Het heeft in zulke gevallen zin, naast (3) nog iets extra's te eisen en dit door een enigszins van (2) afwijkende notatie tot uitdrukking te brengen.
We onderscheiden nu twee gevallen:

a. Bij een gegeven beginvoorwaarde P eisen we dat de eindvoorwaarde Q zo "sterk" mogelijk is. Dit betekent dat voor elke Q' die voldoet aan $\{P\}\ S\ \{Q'\}$, moet gelden $Q \Rightarrow Q'$ (Q impliceert Q'). Voor zo'n eindvoorwaarde Q (die vanzelfsprekend ook aan (2) oftewel (3) voldoet) schrijven we

(5) $\{P\}\ S\ [Q]$.

Er geldt dus wel

$\{x > 5\}\ x := x + 1\ \{x > 0\}$,

$\{x > 5\}\ x := x + 1\ \{x > 6\}$ en

$\{x > 5\}\ x := x + 1\ [x > 6]$, maar niet

$\{x > 5\}\ x := x + 1\ [x > 0]$.

We noemen (5) een voorwaartse regel.

b. Bij een gegeven eindvoorwaarde Q eisen we dat de beginvoorwaarde P zo "zwak" mogelijk is. Dit betekent dat voor elke P' die voldoet aan $\{P'\}\ S\ \{Q\}$, moet gelden $P' \Rightarrow P$. Voor zo'n beginvoorwaarde P (die ook aan (2) oftewel (3) voldoet) schrijven we

(6) $[P]\ S\ \{Q\}$.

Er geldt dan wel

$\{x > 8\}\ x := x + 1\ \{x > 6\}$ en

$[x > 5]\ x := x + 1\ \{x > 6\}$, maar niet

$[x > 8]\ x := x + 1\ \{x > 6\}$.

We noemen (6) een achterwaartse regel.

Er zijn twee bijzondere soorten van voorwaarden. In de eerste plaats zijn er "vanzelfsprekendheden", zoals $x = x$, $2 + 2 = 4$ en $(x+y)(x-y) = x^2 - y^2$. Aan deze voorwaarden is voor alle waarden van x, y enz. voldaan. We beschouwen ze daarom als één voorwaarde, die we noteren als TRUE. Elke voorwaarde impliceert de voorwaarde TRUE. In de tweede plaats zijn er "onmogelijkheden", zoals $x = x + 1$, $2 + 2 = 5$ en $(x+y)(x-y) = x^2 - y^2 + 1$. Aan dit soort voorwaarden is voor geen enkele waarde van x, y enz. voldaan. We vatten dit soort voorwaarden op als één voorwaarde, die we noteren als FALSE. Elke voorwaarde wordt door de voorwaarde FALSE geïmpliceerd. TRUE is de allerzwakste en FALSE de allersterkste voorwaarde. Voor elke voorwaarde P

geldt dus

$$\text{FALSE} \Rightarrow P \Rightarrow \text{TRUE}.$$

We definiëren nu de inherente voorwaarde P_S van statement S door

$$[P_S]\ S\ \{\text{TRUE}\}.$$

P_S is de zwakste voorwaarde waarvoor de betekenis van de statement S is gedefinieerd. Zo is b.v. $x \geq 0$ de inherente voorwaarde van de statement x := sqrt(x). Evenzo is "i geheel en $i \geq 0$" de inherente voorwaarde van de statement

while i ≠ 0 **do** i := i-1 **od**,

zoals men gemakkelijk nagaat. We noemen een beginvoorwaarde sterk genoeg (voor S) als $P \Rightarrow P_S$. Hieruit volgt dat een beginvoorwaarde P sterk genoeg is voor een statement S als en slechts als S gedefinieerd is voor alle toestanden die aan P voldoen. Voor de statement

y := 1/x

is elk van de beginvoorwaarden

$$x = 5, \quad x \neq 0, \quad 0 < x < 3$$

sterk genoeg, maar is geen van de beginvoorwaarden

$$x = 0, \quad -2 < x < 2, \quad \text{TRUE}$$

sterk genoeg. Evenzo is voor de statement

while i ≠ 0 **do** i := i-1 **od**

elk van de beginvoorwaarden

i = 8, "i is een natuurlijk getal"

sterk genoeg in tegenstelling tot elk van de beginvoorwaarden

i = -1, "i is een geheel getal", "i is een positief reëel getal"

Volgens de toelichting bij (2), (3), (5) en (6) kunnen de regels

{P} S {Q} ,

{P} S [Q] en

[P] S {Q}

alleen geldig zijn als P sterk genoeg is voor S, d.w.z. als P impliceert dat de statement S in eindige tijd kan worden uitgevoerd en het resultaat gedefinieerd is.

3. DE ASSIGNMENT STATEMENT

We behandelen hier uitsluitend assignment statements waarvan het linkerlid een ongeïndiceerde variabele is.

a. De voorwaartse regel voor een assignment statement luidt:

Als P sterk genoeg is, dan

(7) $\{P(x)\}\ \ x := \phi(x)\ \ [\exists x_0 : P(x_0) \wedge x=\phi(x_0)]$

("$\wedge$", "$\vee$" en "$\exists x_0$:" betekenen resp. "en", "of" en "er is een x_0 zodanig dat"; bij concrete uitwerkingen van (7) wordt het expliciet schrijven van :$\exists x_0$:" ook wel achterwege gelaten). Regel (7) zal aan de hand van een voorbeeld worden toegelicht. Gevraagd wordt Q te vinden in:

(8) $\{x+y > 0\}\ \ x:=x-y\ \ [Q]$.

In Q moet een uitspraak worden gedaan over de nieuwe waarde die de variabele x door de statement x:=x-y krijgt. Deze nieuwe waarde noteren we ook als x. Voor de oude waarde van de variabele x schrijven

we x_0. De assignment statement geeft nu het volgende verband tussen x en x_0:

(9) $\quad x = x_0 - y$

De getalwaarde x_0 verandert niet door de assignment statement. Daarom geldt

(10) $\quad x_0 + y > 0$

niet alleen vóór deze statement, maar ook erna. Voor Q vinden we dus

(11) $\quad x_0 + y > 0 \ \wedge \ x = x_0 - y.$

De oude waarde x_0 is niet interessant. We elimineren daarom x_0 uit (11) door $x+y$ voor x_0 te substitueren in $x_0+y > 0$. We vinden zo

(12) $\quad \{x+y > 0\} \ \ x := x - y \ \ [x+2y > 0].$

Opmerking. Het is niet altijd eenvoudig x_0 te elimineren, zoals b.v. blijkt uit

$$\{x > \pi\} \ \ x:=x+3*\sin(x) \ \ [x_0 > \pi \ \wedge \ x = x_0 + 3\sin x_0].$$

Hier moet de eindvoorwaarde worden gelezen als: "Er is een getal x_0 zodanig dat

$$x_0 > \pi \ \wedge \ x = x_0 + 3\sin x_0",$$

hetgeen wel degelijk een voorwaarde is die aan x kan worden opgelegd.

b. De achterwaartse regel voor de assignment statement luidt

(13) $\quad [Q(\phi(x))] \ \ x := \phi(x) \ \ \{Q(x)\}.$

Het volgende voorbeeld maakt deze regel duidelijk. Gevraagd wordt P te vinden in

$$[P] \quad x := x*x + y \quad \{x > 10 \ \wedge \ y = 1\}.$$

De waarde van y verandert niet door de assignment statement, dus $y=1$ geldt ook ervóór. Dat achteraf $x > 10$ is, betekent dat de getalwaarde die in het rechterlid van de assignment statement wordt berekend groter dan 10 is. Als we met x de oude waarde bedoelen kunnen we deze getalwaarde schrijven als het rechterlid zelf.
We vinden dus

$$[x^2+y > 10 \ \wedge \ y = 1] \quad x:=x*x+y \quad \{x > 10 \ \wedge \ y = 1\},$$

hetgeen nog kan worden vereenvoudigd tot

$$[|x| > 3 \ \wedge \ y = 1] \quad x:=x*x+y \quad \{x > 10 \ \wedge \ y = 1\}.$$

4. DE CONDITIONELE STATEMENT

a. De voorwaartse regel voor de conditionele statement luidt:

(14) Als $\{P \wedge B\} \ S_1 \ [Q_1]$ en $\{P \wedge \neg B\} \ S_2 \ [Q_2]$, dan

$$\{P\} \ \underline{\text{if}} \ B \ \underline{\text{then}} \ S_1 \ \underline{\text{else}} \ S_2 \ \underline{\text{fi}} \ [Q_1 \vee Q_2].$$

Het volgende voorbeeld verduidelijkt deze regel. Gevraagd wordt Q te bepalen in

(15) $$\{x+y > 0\} \ \underline{\text{if}} \ x > 0 \ \underline{\text{then}} \ x:=x-1 \ \underline{\text{else}} \ y:=y+1 \ \underline{\text{fi}} \ [Q].$$

Als $x:=x-1$ wordt uitgevoerd, weten we niet alleen dat vooraf $x+y > 0$, maar ook dat vooraf $x > 0$. Evenzo gebruiken we bij de statement $y:=y+1$ de extra beginwvoorwaarde $x \leq 0$. Uit de voorwaartse regel (7) voor de assignment statement volgt:

$$\{x+y > 0 \ \wedge \ x > 0\} \quad x:=x-1 \quad [x+y > -1 \ \wedge \ x > -1] \quad \text{en}$$

$$\{x+y > 0 \ \wedge \ x \leq 0\} \quad y:=y+1 \quad [x+y > 1 \ \wedge \ x \leq 0] \ .$$

Voor Q in (15) vinden we zo de voorwaarde

$$(x+y > -1 \ \wedge \ x > -1) \ \vee \ (x+y > 1 \ \wedge \ x \leq 0).$$

b. De achterwaartse regel voor de conditionele statement luidt:

(16) Als $[P_1]\ S_1\ \{Q\}$ en $[P_2]\ S_2\ \{Q\}$, dan

$$[(P_1 \wedge B) \vee (P_2 \wedge \neg B)]\ \underline{\text{if}}\ B\ \underline{\text{then}}\ S_1\ \underline{\text{else}}\ S_2\ \underline{\text{fi}}\ \{Q\}.$$

Het volgende voorbeeld verduidelijkt deze regel.
Gevraagd wordt P te bepalen in

$$[P]\ \underline{\text{if}}\ x > 0\ \underline{\text{then}}\ x:=x-1\ \underline{\text{else}}\ y:=y+1\ \underline{\text{fi}}\ \{x+y = 100\}.$$

Uit de achterwaartse regel (13) voor de assignment statement volgt

$$[x+y = 101]\ \ x:=x-1\ \ \{x+y = 100\} \quad \text{en}$$

$$[x+y = 99\ ]\ \ y:=y+1\ \ \{x+y = 100\}.$$

Als x:=x-1 wordt uitgevoerd geldt ervóór $x > 0$;
als y:=y+1 wordt uitgevoerd geldt ervóór $x \leq 0$.
De gezochte uitdrukking van P is daarom

$$(x+y = 101 \ \wedge \ x > 0) \vee (x+y = 99 \ \wedge \ x \leq 0).$$

5. DE SAMENGESTELDE STATEMENT

Als S en T statements zijn, dan vatten we

S;T

op als één nieuwe statement, een z.g. samengestelde statement. In deze samengestelde statement wordt eerst S en daarna T uitgevoerd.

a. De voorwaartse regel voor de samengestelde statement luidt:

(17) Als $\{P\}\ S\ [Q]$ en $\{Q\}\ T\ [R]$, dan

$\{P\}\ S;T\ [R]$.

b. De achterwaartse regel voor de samengestelde statement luidt:

(18) Als $[P]\ S\ \{Q\}$ en $[Q]\ T\ \{R\}$, dan

$[P]\ S;T\ \{R\}$.

6. DE WHILE-STATEMENT

Als voor S de while-statement

$$\underline{\text{while}}\ B\ \underline{\text{do}}\ T\ \underline{\text{od}}$$

wordt genomen, is het formuleren en toepassen van de regels $\{P\}\ S\ [Q]$ en $[P]\ S\ \{Q\}$ aanzienlijk bewerkelijker dan wat we tot nu toe hebben gezien. Daarom volgt nu eerst de eenvoudiger regel:

(19) Als $\{P \wedge B\}\ T\ \{P\}$ en als P sterk genoeg (voor de while-statement) is, dan

$$\{P\}\ \underline{\text{while}}\ B\ \underline{\text{do}}\ T\ \underline{\text{od}}\ \{P \wedge \neg B\}.$$

Voor het toepassen van regel (19) moet men zelf een geschikt "invariante voorwaarde" P vinden. Laten we ons bijvoorbeeld ten doel stellen te bewijzen dat x door de statement

(20) $\underline{\text{while}}\ i>0\ \underline{\text{do}}\ x:=x*a;\ i:=i-1\ \underline{\text{od}}$

gelijk wordt gemaakt aan a^n als gegeven is dat vooraf geldt

$$i = n \geq 0 \quad \wedge \quad x = 1 \quad \text{(i en n zijn gehele getallen)}.$$

We moeten nu een zinvolle voorwaarde P opsporen waarvoor geldt

(21) $\quad \{P \wedge i>0\} \quad x:=x*a \quad \{P\}.$

Komt men op de gelukkige gedachte voor P de voorwaarde $x=a^{n-i} \wedge i\geq 0$ te nemen, dan blijkt regel (19) bijzonder prettig bruikbaar te zijn. We moeten nu verifiëren of (21) voor deze P geldt, m.a.w. of

$$\{x=a^{n-i} \wedge i>0\} \quad x:=x*a;\ i:=i-1 \quad \{x=a^{n-i} \wedge i\geq 0\}.$$

Nu levert achtereenvolgende toepassing van de achterwaartse regel (13) voor de assignment statement:

$$[x=a^{n-i+1} \wedge i\geq 1] \quad i:=i-1 \quad \{x=a^{n-i} \wedge i\geq 0\}$$
$$[x=a^{n-i} \wedge i\geq 1] \quad x:=x*a \quad \{x=a^{n-i+1} \wedge i\geq 1\}.$$

Volgens de achterwaartse regel (18) voor de samengestelde statement geldt dan

$$[x=a^{n-i} \wedge i\geq 1] \quad x:=x*a;\ i:=i-1 \quad \{x=a^{n-i} \wedge i\geq 0\}.$$

Omdat i geheel is, is $i\geq 1$ identiek met $i>0$; verder volgt i.h.a. uit $[P]\ S\ \{Q\}$, dat ook $\{P\}\ S\ \{Q\}$. Dus blijkt (21) inderdaad voor de gekozen P te gelden. Volgens regel (19) geldt dan dat na de while-statement voldaan is aan $P \wedge \neg B$, hetgeen hier

$$(x = a^{n-i} \quad \wedge \quad i \geq 0) \quad \wedge \quad i \leq 0, \quad \text{oftewel}$$
$$x = a^{n}$$

betekent.
We zullen nu ook de wat gecompliceerdere regels

$\{P\}$ **while** B **do** T **od** $[Q]$ en

$[P]$ **while** B **do** T **od** $\{Q\}$

formuleren.

a. De voorwaartse regel voor de while-statement luidt:

(22) Als P sterk genoeg is, dan

$\{P\}$ **while** B **do** T **od** $[\neg B \wedge (U_0 \vee U_1 \vee U_2 \vee \ldots)]$,

waarin U_k gegeven is door

$U_0 \equiv P$

$\{B \wedge U_{k-1}\}$ T $[U_k]$ $(k=1,2,\ldots)$.

b. De achterwaartse regel voor de while-statement luidt:

(23) $[V_0 \vee V_1 \vee V_2 \vee \ldots]$ **while** B **do** T **od** $\{Q\}$,

waarin V_k als volgt is gegeven

$V_0 \equiv (\neg B \wedge Q)$

$V_k \equiv (B \wedge W_k)$ $(k=1,2,\ldots)$,

waarbij W_k uit V_{k-1} gevonden wordt volgens

$[W_k]$ T $\{V_{k-1}\}$.

Uit het volgende voorbeeld moge de praktische bruikbaarheid van (22) blijken. We passen deze regel toe op de eerder besproken while-statement

while i>0 **do** x:=x*a; i:=i-1 **od**,

waarbij we uitsluitend uitgaan van de gegeven beginvoorwaarde $i=n\geq 0 \wedge x=1$; deze is sterk genoeg. We hoeven nu dus niet zelf een invariante voorwaarde op te sporen. Volgens (22) hebben we dan

$U_0 \equiv P \equiv (i=n\geq 0 \wedge x=1)$.

Eveneens volgens (22) wordt U_k hier uit U_{k-1} gevonden volgens:

(24) $\quad \{i>0 \wedge U_{k-1}\}\ \ x:=x*a;\ i:=i-1\ \ [U_k].$

We vinden daarom U_1 uit

$$\{i>0 \wedge i=n \wedge x=1\}\ \ x:=x*a;\ i:=i-1\ \ [U_1],$$

waaruit volgt: $U_1 \equiv (i\geq 0 \wedge i+1=n \wedge x=a)$.
Vervolgens kan U_2 volgens (24) worden gevonden uit

$$\{i>0 \wedge i+1=n \wedge x=a\}\ \ x:=x*a;\ i:=i-1\ \ [U_2],$$

hetgeen oplevert: $U_2 \equiv (i\geq 0 \wedge i+2=n \wedge x=a^2)$.
Evenzo vinden we $U_3 \equiv (i\geq 0 \wedge i+3=n \wedge x=a^3)$, enz.
De te bepalen eindvoorwaarde is volgens (22):

(25) $\quad \neg B \wedge (U_0 \vee U_1 \vee U_2 \vee \ldots).$

Nu is $(\neg B) \equiv (i\leq 0)$. Anderzijds impliceert elke U_k dat $i \geq 0$. Hieruit volgt $i = 0$. We substitueren daarom 0 voor i in alle U_k bij het uitschrijven van (25) en vinden:

$$\begin{aligned} i = 0 \ \wedge\ [&(0 = n \ \wedge\ x = 1) \vee \\ &(1 = n \ \wedge\ x = a) \vee \\ &(2 = n \ \wedge\ x = a^2) \vee \ldots]. \end{aligned}$$

Dit is identiek met $i = 0 \ \wedge\ n \geq 0 \ \wedge\ x = a^n$.
We hebben nu gevonden:

$$\{i=n\geq 0 \wedge x=1\}\ \ \underline{\text{while}}\ i \neq 0\ \underline{\text{do}}\ x:=x*a;\ i:=i-1\ \underline{\text{od}}$$
$$[i=0 \wedge n\geq 0 \wedge x=a^n]\ .$$

7. ENIGE EIGENSCHAPPEN VAN BEWIJSREGELS

Er zijn enkele interessante eigenschappen van bewijsregels te vermelden. In [10] wordt met behulp van enige begrippen en stellingen uit de verzamelingenleer duidelijk gemaakt hoe men deze eigenschappen kan bewijzen. Er is een nauw verband tussen een voorwaarde P en een zekere deelverzameling X van een "toestandsruimte" V; X bestaat juist uit die punten $\underline{x}$ uit V die aan de voorwaarde P voldoen. Met een statement S wordt een functie f geassocieerd, zodanig dat {P} S [Q] wordt beschreven door:

$$f(X) = Y,$$

waarin Y de deelverzameling van V is die juist uit die punten $\underline{y}$ bestaat die voldoen aan Q. Evenzo is

$$f^{-1}(Y) = X$$

equivalent met [P] S {Q}. Dezelfde eigenschappen die in [10] met behulp van verzamelingen en functies worden behandeld, worden hier uitgedrukt met behulp van voorwaarden en statements. Eigenschap a, zal met een voorbeeld worden toegelicht. De lezer wordt aangeraden zelf concrete voorbeelden van de overige eigenschappen te bedenken.

a. Uit {P} S [Q] volgt {P} S {Q}.
   Laat {P} S [Q] slechts één voorwaarde P toe, dan volgt uit {P} S [Q] zelfs [P] S {Q}.
   (zie onderstaande toelichting)

b. Uit [P] S {Q} volgt {P} S {Q}.
   Geldt $\{P_S\}$ S [TRUE], dan volgt uit [P] S {Q} zelfs {P} S [Q].

c. Zij gegeven: $\{P_1\}$ S $[Q_1]$ en $\{P_2\}$ S $[Q_2]$. Er geldt dan:
   c1. Uit $P_1 \Rightarrow P_2$ volgt $Q_1 \Rightarrow Q_2$.
   c2. $\{P_1 \vee P_2\}$ S $[Q_1 \vee Q_2]$.

c3. $\{P_1 \wedge P_2\}\ S\ \{Q_1 \wedge Q_2\}$.

Laat $\{P\}\ S\ [Q]$ slechts één voorwaarde P toe, dan geldt zelfs:

$\{P_1 \wedge P_2\}\ S\ [Q_1 \wedge Q_2]$.

d. Zij gegeven: $[P_1]\ S\ \{Q_1\}$ en $[P_2]\ S\ \{Q_2\}$.
Er geldt dan:

d1. Uit $Q_1 \Rightarrow Q_2$ volgt $P_1 \Rightarrow P_2$.

d2. $[P_1 \vee P_2]\ S\ \{Q_1 \vee Q_2\}$.

d3. $[P_1 \wedge P_2]\ S\ \{Q_1 \wedge Q_2\}$.

e. De volgende drie uitspraken zijn (omdat aan $P \Rightarrow P_S$ voldaan is) gelijkwaardig:

e1. $\{P\}\ S\ [FALSE]$,

e2. $\{P\}\ S\ \{FALSE\}$ en

e3. $P \equiv FALSE$.

f. Uit $[P]\ S\ \{Q\}$ volgt $[P_S \wedge \neg P]\ S\ \{\neg Q\}$.

De onder a. genoemde eigenschap wordt door het volgende voorbeeld toegelicht.

Uit $\{x = 3\}\ x:=x*x\ [x = 9]$ volgt uiteraard

$\{x = 3\}\ x:=x*x\ \{x = 9\}$.

Uit $\{x = 3\}\ x:=x*x*x\ [x = 27]$ volgt zelfs

$[x = 3]\ x:=x*x*x\ \{x = 27\}$.

Het verschil bestaat daarin, dat

$\{P\}\ x:=x*x\ [x = 9]$

twee voorwaarden P toelaat, n.l. $x = 3$ en $x = -3$, terwijl

$\{P\}\ x:=x*x*x\ [x = 27]$

slechts één voorwaarde P toelaat, n.l. $x = 3$.

LITERATUUR

[1] FLOYD, R.W., *Assigning Meanings to Programs*, Proc. Symp, Appl Math. 19, American Math. Soc. (1967) 19-32.

[2] HOARE, C.A.R., *An axiomatic Basis of Computer Programming*, CACM, Vol. 12. No. 10 (October 1969), 576-580.

[3] DIJKSTRA, E.W., *A Simple Axiomatic Basis for Programming Language constructs*, Proc. Kon. Ned. Akad., Ser. A, 77 (or Indagationes Math., 36), 1-15 (1974).

[4] MANNA, Z. & A. PNUELI, *Axiomatic Approach to Total Correctness of Programs*, Report STAN-CS-73-382, Stanford University (1973).

[5] DE BAKKER, J.W., *Flow of control in the proof theory of structured programming*, Proc. 16th IEEE Symp. on Foundations of Computer Science (1975).

[6] MILLS, H.D., *The New Math of Computer Programming*, CACM, Vol. 18, No. 1 (January 1975), 43-48.

[7] NAUR, P., *An Experiment on Program Development*, BIT, 12 (1972), 347-365.

[8] WIRTH, N., *Systematisches Programmieren*, Teubner, 1972.

[9] DE BAKKER, J.W. (red.), *Colloquium Programmacorrectheid*, MC Syllabus 21, Mathematisch Centrum, Amsterdam, 1975.

[10] AMMERAAL, L., *How program statements transform predicates*, Prepublication, Rapport IW 39/75, Mathematisch Centrum, Amsterdam, 1975.