MASTER

Realisering van sequentiële machines met behulp van de partitie-theorie

Jansen, P.G.

Award date:
1972

ECB-471

AFDELING DER ELEKTROTECHNIEK

TECHNISCHE HOGESCHOOL

EINDHOVEN

Groep E.C.B.

<u>REALISERING VAN SEQUENTIËLE</u>

<u>MACHINES MET BEHULP VAN DE</u>

<u>PARTITIE-THEORIE</u>

Door P.C. Jansen

Rapport van het afstudeerwerk

uitgevoerd van maart 1970 tot dec. 1970

in opdracht van prof. ir. A. Heetman

onder leiding van ir. A.J. Ekas

Samenvatting.

Het gedrag van een sequentiele machine kan worden vastgelegd in een volgorde tabel.
Met behulp van deze tabel is het mogelijk een ontwerp te geven voor de opbouw van deze machines.
Als hulpmiddel voor het verkrijgen van een gunstig ontwerp is gebruikgemaakt van de Algebraische Structuur Theorie van sequentiele machines.
Met deze theorie kunnen de voor het ontwerp bruikbare Mm-paren worden verkregen.
Dit rapport zal achtereenvolgens beschrijven:

1) Een overzicht van de algebraische structuur theorie.
2) Een methode voor het verkrijgen van Mm-paren.
3) Een ontwerpmethode voor sequentiele machines waarbij de geheugenelementen beperkt worden tot de groep van binaire geheugenelementen.
4) Een ontwerp waarbij de geheugenelementen bestaan uit j-k-flipflops.

Inhoud.

Inleiding.

Bij analyse en ontwerp van sequentiele machines kan op zinvolle wijze gebruik worden gemaakt van de partitie-theorie.

Met behulp van deze theorie wordt een verzameling toestanden van een sequentiele machine onderverdeeld in verschillende deelverzamelingen van toestanden.

Indien deze deelverzamelingen op een geschikte manier worden gekozen, geven ze richtlijnen voor het opsplitsen van een machine in verschillende deelmachines.

Via deze richtlijnen is het in vele gevallen mogelijk te komen tot een zo economisch mogelijke opbouw van een sequentiele machine.

1. De sequentiele machine.

We maken onderscheid tussen twee soorten machines en geven een definitie van beide soorten:

Definitie 1.1

Een Moore-machine is een 5-tuple $M=(S,I,O,\delta,\lambda)$ waarbij:

a) S een eindige niet lege verzameling van toestanden is.

b) I een eindige niet lege verzameling van inputs is.

c) O een eindige niet lege verzameling van outputs is.

d) δ de overgangsfunctie is die de machine verkerende in toestand s_1, bij input i_1 naar de toestand s_2 doet gaan. In verkorte notatie $\delta(s_1,i_1)=s_2$.

e) λ is de uitgangsfunctie. Bij een bepaalde toestand s_1 hoort een output o_1; dan is de notatie: $\lambda(s_1)=o_1$.

definitie 1.2

Een mealy-machine is een 5-tuple $M=(S,I,O,\delta,\beta)$ waarbij:

a) t.m. d) dezelfde eigenschappen als de Moore-machine.

e) β is de uitgangsfunctie. Indien een machine verkeert in een toestand s_1 en een input i_1 krijgt toegevoegd, dan is gedurende de overgang van s_1 naar s_2 ($\beta(s_1,i_1)=s_2$) de output o_1. In verkorte notatie: $\beta(s_1,i_1)=o_1$. Indien $\beta(s_1,i_1)=\lambda(s_1)$ dan gaat de Mealy-machine over in de Moore-machine.

Voor fig. 1 geldt:

$M=(S,I,O,\delta,\lambda)$ met

$S=(1,2,3)$; $I=(a,b)$; $O=(0,1)$.

en $\delta(1,a)=2$; $\delta(2,b)=1$; $\lambda(3)=0$; $\lambda(2)=1$.

inputs

toestanden	a	b	outputs
1	2	3	0
2	3	1	1
3	1	2	0

fig. 1. Voorbeeld van een Moore- machine.

Voor fig. 2 geldt $M=(S,I,O,\delta,\beta)$;

met $S=(1,2,3)$; $I=(a,b)$; $O=(0,1)$;

en $\delta(1,b)=3$; $\delta(3,a)=1$;

en $\beta(2,b)=0$; $\beta(2,a)=1$; $\beta(3,b)=1$.

inputs

toestanden	a	b	a (outputs)	b (outputs)
1	2	3	0	0
2	3	1	1	0
3	1	2	0	1

fig. 2. Voorbeeld van een Mealy-machine.

2. Definities en eigenschappen t.a.v. partities.

definitie 2.1.

Een partitie π bestaat uit een verzameling van disjuncte deelverzamelingen B_i van een verzameling S, zodanig dat de vereniging van alle B_i weer de verzameling S oplevert.

Voorbeeld:

$$S=(1,2,3,4,5,6,7,8)$$

$$B_1=(1,2,3,5);\ B_2=(4,6,7);\ B_3=(8);$$

dan is $\pi=((1,2,3,5),(4,6,7),(8))$.

De gebruikelijke notatie is: $\pi=(\overline{1,2,3,5};\overline{4,6,7};\overline{8})$.

Indien twee elementen in hetzelfde blok B_i voorkomen, dan noteert men dit als volgt:

$1\equiv 2(\pi)$; of $4\equiv 6(\pi)$; algemeen: $s\equiv t(\pi)$.

definitie 2.2.

Voor de doorsnede van twee partities $\pi_1.\pi_2=\pi_3$ geldt:

$s\equiv t(\pi_3)$ dan en alleen dan als $s\equiv t(\pi_1)$ en $s\equiv t(\pi_2)$.

definitie 2.3.

Voor de vereniging van twee partities $\pi_1+\pi_2=\pi_4$ geldt:

$s\equiv t(\pi_4)$ dan en alleen dan indien er een aantal elementen van S bestaan: $s=s_0,s_1,s_2,\ldots.s_n=t$ die voldoen aan een van de volgende twee relaties:

$s_i\equiv s_{i+1}(\pi_1)$ of $s_i\equiv s_{i+1}(\pi_2)$ met $0\leq i\leq n-1$.

Voorbeelden: beschouw de verzameling $S=(1,2,3,4,5,6,7,8,9)$.

met $\pi_1=(\overline{1,2};\overline{3,4};\overline{5,6};\overline{7,8,9})$ en $\pi_2=(\overline{1,6};\overline{2,3};\overline{4,5};\overline{7,8};\overline{9})$ dan is:

$\pi_3=\pi_1.\pi_2=(\overline{1};\overline{2};\overline{3};\overline{4};\overline{5};\overline{6};\overline{7,8,9})$ en $\pi_4=\pi_1+\pi_2=(\overline{1,2,3,4,5,6};\overline{7,8,9})$.

definitie 2.4.

Een partitie $\pi_1\leq\pi_2$ of $\pi_2\geq\pi_1$ indien ieder blok van π_1 bevat is in een blok van π_2.

Voorbeeld: $S=(1,2,3,4,5)$.

$\pi_1=(\overline{B_1},\overline{B_2},\overline{B_3})=(\overline{1,2};\overline{3,4};\overline{5})$ en $\pi_2=(\overline{B_4},\overline{B_5})=(\overline{1,2};\overline{3,4,5})$
dan $B_1\subseteq B_4$ en $B_2\subseteq B_5$ en $B_3\subseteq B_5$.

Ieder blok van π_1 ligt dus in een blok van π_2, dus $\pi_1\leq\pi_2$.

De kleinst mogelijke partitie op S noemen we de nul-partitie.

Notatie: $0=(\overline{1};\overline{2};\overline{3};\overline{4};\overline{5})$; alle toestanden komen in een verschillend blok voor.

De grootst mogelijke partitie op S noemen we de eenheids-partitie.

Notatie: $I=(\overline{1,2,3,4,5})$; alle toestanden komen in hetzelfde blok voor.

definitie 2.5.

a) De kleinste bovengrens (least upperbound) van twee partities π_1 en π_2 is een partitie π_3 waarvoor geldt:

   1) $\pi_1\leq\pi_3$

   2) $\pi_2\leq\pi_3$

   3) Indien $\pi_1\leq\pi_4$ en $\pi_2\leq\pi_4$ dan geldt ook $\pi_3\leq\pi_4$.

notatie: $\pi_3=\text{l.u.b.}(\pi_1,\pi_2)$

b) De grootste ondergrens (greatest lower bound) van twee partities π_1 en π_2 is een partitie π_3 waarvoor geldt:

   1) $\pi_1\geq\pi_3$

   2) $\pi_2\geq\pi_3$

   3) Indien $\pi_1\geq\pi_4$ en $\pi_2\geq\pi_4$ dan geldt ook $\pi_3\geq\pi_4$.

notatie: $\pi_4 = $ g.l.b. (π_1, π_2).

Uit def. 2.2 en def. 2.5a volgt dat indien $\pi_3 = $ l.u.b. (π_1, π_2) dan $\pi_3 = \pi_1 + \pi_2$, en uit def.2.3 en def. 2.5b volgt dat indien $\pi_3 = $ g.l.b. (π_1, π_2) dan $\pi_3 = \pi_1 . \pi_2$.

Uit def. 4,5 en6 zijn de volgende eigenschappen af te leiden:

1) $\pi_1 . \pi_2 = \pi_1$ indien $\pi_1 \leqslant \pi_2$;

$\pi_1 + \pi_2 = \pi_1$ indien $\pi_1 \geqslant \pi_2$;

2) $\pi_1 . \pi_2 = \pi_2 . \pi_1$ (commutatief)

$\pi_1 + \pi_2 = \pi_2 + \pi_1$

3) $\pi_1(\pi_2 . \pi_3) = (\pi_1 . \pi_2)\pi_3$ (associatief)

$\pi_1 + (\pi_2 + \pi_3) = (\pi_2 + \pi_3) + \pi_1$

4) $\pi_1(\pi_1 + \pi_2) = \pi_1$

$\pi_1 + (\pi_1 . \pi_2) = \pi_1$

5) $\pi_1(\pi_2 + \pi_3) \geqslant \pi_1 . \pi_2 + \pi_1 . \pi_3$

$\pi_1(\pi_2 . \pi_3) \leqslant (\pi_1 + \pi_2)\ (\pi_1 + \pi_3)$

Op het eerste gezicht zou men bij de 5^e eigenschap een gelijkheid verwachten.
Het volgende voorbeeld toont echter aan dat de ongelijkheid wel op zijn plaats is.
voorbeeld:
$S = (1,2,3)$; $\pi_1 = (\overline{1,2}; \overline{3})$; $\pi_2 = (\overline{1}; \overline{2,3})$; $\pi_3 = (\overline{1,3}; \overline{2})$.
dan is $\pi_1(\pi_2 + \pi_3) = \pi_1$ en $\pi_1 . \pi_2 = 0$ en $\pi_1 . \pi_3 = 0$ dus $\pi_1 . \pi_2 + \pi_1 . \pi_3 = 0$
dus $\pi_1(\pi_2 + \pi_3) > \pi_1 . \pi_2 + \pi_1 . \pi_3$.
Verder is $\pi_1 + (\pi_2 . \pi_3) = \pi_1$ en $(\pi_1 + \pi_2)\ (\pi_1 + \pi_3) = I$ dus $\pi_1 + (\pi_2 . \pi_3) < (\pi_1 + \pi_2)\ (\pi_1 + \pi_3)$.

3.0. Definities en eigenschappen t.a.v. partitieparen.

definitie 3.1.

Het partitiepaar (π_1,π_2) van de machine $M=(S,I,O,\delta,\lambda)$ met $S=(s_1,s_2,\ldots.s_k)$ en $I=(i_1,i_2,\ldots.i_m)$ is een geordend paar van partities op S zodat voor $s_i \equiv s_j\ (\pi_1)$ geldt dat $\delta(s_i,x) \equiv \delta(s_j,x)\ (\pi_2)$ waarbij $x \in I$.

Voorbeeld:

Beschouw de volgorde tabel van machine A.

	a	b	c	d
1	1	2	3	4
2	3	4	1	2
3	2	1	4	3
4	4	3	2	1

fig. 2. Machine A

Voor deze machine zijn:

$(\pi_1,\pi_2)=((\overline{1,2};\overline{3,4}),(\overline{1,3};\overline{2,4}))$
$(\pi_2,\pi_1)=((\underline{1,3};\underline{2,4}),(\underline{1,2};\underline{3,4}))$ en
$(\pi_3,\pi_3)=((1,4;2,3),(1,4;2,3))$

partitieparen.

Merk op dat van het laatste partitiepaar (π_3,π_3) beide partities gelijk zijn.

Indien dit het geval is dan heet π_3 een partitie met <u>substitutie eigenschap</u>.

Lemma 3.1.

Indien (π_1,π_2) en (τ_1,τ_2) partitieparen zijn van een sequentiele machine M dan zijn

1) $((\pi_1 \cdot \tau_1),(\pi_2 \cdot \tau_2))$ en

2) $((\pi_1+\tau_1),(\pi_2+\tau_2))$ ook partitieparen van de machine M.

Het bewijs wordt geleverd door Hartmanis en Stearns[1].

Beschouw alle partitieparen van een machine M $[(\pi,\pi_1)\ldots(\pi,\pi_k)]$ waarbij π een gegeven partitie is.

$\prod(\pi_i)=(\pi_1.\pi_2\ldots\pi_i\ldots\pi_k)=\pi'$

Dan wordt de doorsnede van alle rechter partities die met π een partitiepaar van machine M vormen genoteerd als:

$\pi'=\prod[\pi_i|(\pi,\pi_i)$ is een partitiepaar van M]

Indien $\sum(\pi_i)=(\pi_1+\pi_2+\ldots+\pi_i+\ldots\pi_k)=\pi''$,

dan wordt de som van alle linker partities die met π een partitiepaar van de machine M vormen, genoteerd als:

$\pi''=\sum[\pi_i|(\pi_i,\pi)$ is een partitiepaar van M].

definitie 3.2.

Indien π een partitie is behorende bij de machine M dan is:

$m(\pi)=\prod[\pi_i|(\pi,\pi_i)$ is een partitiepaar van M].

en

$M(\pi)=\sum[\pi_i|(\pi_i,\pi)$ is een partitiepaar van M].

Er volgen nu een aantal eigenschappen van partitieparen:

lemma 3.2.

Als (π,τ) een partitiepaar is van de machine M, $\tau\geq\tau'$ en $\pi'\leq\pi$ dan zijn (π',τ); (π,τ') en (π',τ') ook partitieparen van de machine M.

Het bewijs wordt geleverd door Hartmanis en Stearns[1].

Lemma 3.2 betekent dat men de linkse partitie mag verkleinen en de rechtse partitie mag vergroten; het resultaat zal weer een partitiepaar zijn.

<u>Definitie 3.3</u>.

Het partitiepaar (π,τ) wordt een Mm-paar genoemd indien:
$\pi=M(\tau)$ en $\tau=m(\pi)$.

<u>stelling 3.1</u>.

Indien (π,τ) een partitiepaar van machine M is dan volgt:

1) $(M(\tau),\tau)$ en $(\pi,m(\pi))$ zijn p.p. van M;

2) $\pi_1\geqslant\pi_2$ houdt in dat $m(\pi_1)\geqslant m(\pi_2)$;

3) $m(\pi_1+\pi_2)=m(\pi_1)+m(\pi_2)$;

4) $m(\pi_1.\pi_2)\leqslant m(\pi_1).m(\pi_2)$;

5) $\tau\geqslant m(\pi)$ dan en alleen dan als (π,τ) een p.p. van M is;

6) $\tau_1\geqslant\tau_2$ houdt in dat $M(\tau_1)$ $M(\tau_2)$;

7) $M(\tau_1+\tau_2)\geqslant M(\tau_1)+M(\tau_2)$;

8) $M(\tau_1.\tau_2)=M(\tau_1).M(\tau_2)$;

9) $\pi\leqslant M(\tau)$ dan en alleen dan als (π,τ) een p.p. van M is;

10) $M(m(\pi))\geqslant\pi$;

11) $m(M(\tau))\leqslant\tau$;

12) $M(m(M(\tau)))=M(\tau)$;

13) $m(M(m(\pi)))=m(\pi)$;

14) $(M(\tau),m(M(\tau)))$ en $(M(m(\pi))),m(\pi))$ zijn partitieparen van M;

15) Indien (π_1,τ_1) en (π_2,τ_2) partitieparen van M zijn dan is $\pi_1\leqslant\pi_2$ dan en alleen dan als $\tau_1\leqslant\tau_2$.

16) Indien (π_1,τ_1) en (π_2,τ_2) partitieparen van M zijn dan geldt: $g.l.b.[(\pi_1,\tau_1),(\pi_2,\tau_2)]=((\pi_1.\pi_2),m(\pi_1.\pi_2))$;

$l.u.b.[(\pi_1,\tau_1),(\pi_2,\tau_2)]=(M(\tau_1+\tau_2),(\tau_1+\tau_2))$.

De bewijzen worden geleverd door Hartmanis en Stearns[1].

Uit de definities en eigenschappen van de partitieparen kan nu een algorithme worden afgeleid voor het verkrijgen van Mm-paren.
Voordat we overgaan tot de beschrijving van het algorithme zullen we echter eerst nagaan welk nut deze Mm-paren hebben bij een toestandstoekenning van een sequentiele machine M.
Deze toekenning is gebaseerd op de hoofdstelling (3.2).
Voor de formulering volgt eerst een korte inleiding.
Veronderstel dat de toestanden van een sequentiele machine worden toegewezen aan de variabelen y_1 tm. y_k.
De bijbehorende partities zijn dan $\pi(y_i)$.
Voorbeeld: S=(1,2,3,4,5,6)
Een toekenning kan dan zijn:

	y_1	y_2	y_3
1	0	0	0
2	1	0	0
3	0	1	0
4	1	1	0
5	0	1	1
6	1	1	1

dan is $\pi(y_1) = (\overline{1,3,5};\overline{2,4,6})$ en

$\pi(y_2) = (\overline{1,2};\overline{3,4,5,6})$ en

$\pi(y_3) = (\overline{1,2,3,4};\overline{5,6})$.

Na de toekenning zijn de secundaire variabelen Y_i een functie van de primaire variabelen $(y_1,y_2,\ldots y_k)$ en van de input $(x_1,x_2,\ldots x_m)$.
Algemeen:

$$Y_i=f(y_1,\ldots y_k,x_1,\ldots x_m).$$

Nu doet zich de mogelijkheid voor dat de betreffende Y_i niet afhankelijk is van alle $y_{i(i=1\text{ tm }k)}$ doch van een deelverzameling daarvan. (bijv. $(y_1,y_3,y_4,\ldots y_{k-2},y_k)$)

De verzameling van bijbehorende indices van deze deel= verzameling noemen we P_i.
(bij het eerder genoemde voorbeeld: $P_i=(1,3,4,\ldots k-2,k)$).
We zijn nu in staat het eerste gedeelte van de hoofd- stelling te formuleren:

Stelling 3.2.A

Veronderstel dat de toestanden van de machine M zijn toe- gewezen aan de variabelen $y_1 \ldots . y_k$, met als bijbehorende partities $\tau(y_1)\ldots.\tau(y_k)$.
Indien v_i slechts afhangt van de input $(x_1\ldots.x_m)$ en van $(v_j \mid j\in P_i)$ dan volgt met $\pi_i = M(\tau(y_i))$:

$$\prod_{j\in P_i} \tau(y_j) \leq \pi_i$$

De stelling zal geillustreerd worden aan de hand van het volgende voorbeeld:

toewijzing:

	00	01	11	10
1	3	1	4	2
2	1	5	4	2
3	3	4	3	5
4	5	1	4	2
5	5	4	3	5

	v_1	v_2	v_3
1	0	0	0
2	1	1	0
3	0	1	1
4	0	1	0
5	0	0	1

fig. 3. Machine B met toewijzing.

Uit de toewijzingstabel volgt:

$\tau(v_1)=(\overline{1,3,4,5};\overline{2})$ $\qquad M(\tau(v_1))=(\overline{1,2,4};\overline{3,5})$

$\tau(v_2)=(\overline{1,5};\overline{2,3,4})$ $\qquad M(\tau(v_2))=(\overline{1};\overline{2,4};\overline{3};\overline{5})$

$\tau(v_3)=(\overline{1,2,4};\overline{3,5})$ $\qquad M(\tau(v_3))=(\overline{1,4};\overline{3,5};\overline{2})$

De logische vergelijkingen met de betreffende toewijzingen zijn dan:

$Y_1=x_1\bar{x}_2\bar{y}_3$ $\quad Y_1=f(y_3,x_1,x_2)$

$Y_2=x_1\bar{y}_3+x_2y_3+\bar{x}_1y_2y_3+\bar{x}_2\bar{y}_2\bar{y}_3$ $\quad Y_2=f(y_2,y_3,x_1,x_2)$

$Y_3=x_1y_3+\bar{x}_1\bar{x}_2\bar{y}_1+\bar{x}_1x_2y_1$ $\quad Y_3=f(y_1,x_1,x_2)$

Voor i=1 volgt voor P_i: $P_i=(3)$.

$\prod_{j\in P_i}\tau(y_j)=\tau(y_3)\leq M(\tau(y_1))$ hetgeen gemakkelijk te verifiëren valt.

Voor i=2 volgt $P_i=(2,3)$.

Dan is $\prod_{j\in P_i}\tau(y_j)=\tau(y_2).\tau(y_3)=(\overline{1};\overline{2,4};\overline{3};\overline{5})$.

$M(\tau(y_2))= \quad (\overline{1};\overline{2,4};\overline{3};\overline{5})$.

Ook hier is de stelling weer geverifiëerd.

Voor i=3 is de stelling eveneens eenvoudig te controleren.

Het tweede gedeelte van de hoofdstelling is het omgekeerde van het eerste gedeelte.

Voor de formulering volgt eerst een korte inleiding.

Stel er zijn een aantal partitieparen $((\pi_1,\tau_1)\ldots.(\pi_k,\tau_k))$.

Met behulp van de rechtse partities τ_i worden de toestanden aan de secundaire variabelen Y_i toegewezen.

Voor de machine B is$((\overline{1,4};\overline{2};\overline{3,5}),(\overline{1,4};\overline{2};\overline{3,5}))$ een partitiepaar.

Maken we bijvoorbeeld gebruik van flipflops dan hebben we minstens twee variabelen Y nodig om de verschillende blokken van de partitie $(\overline{1,4};\overline{2};\overline{3,5})$ te onderscheiden.

Dit kan bv. op de volgende manier:

	y_i	y_j
1	0	0
2	1	0
3	0	1
4	0	0
5	0	1

Voor partities met meerdere blokken hebben we dus ook meer variabelen Y nodig om een toekenning te doen. De deelverzameling van indices van Y behorende bij τ_j noemen we Q_j.

We zijn nu in staat het tweede gedeelte van de stelling te formuleren.

Stelling 3.2.B.

Stel er zijn een aantal partitieparen $((\pi_1,\tau_1)\ldots.(\pi_k\tau_k))$ van een machine M.

Voor iedere i is er dan een verzameling van indices P_i zodanig dat:

$$\prod_{j\in P_i} \tau_j \leq \pi_i$$

Er bestaat dan een toestandstoewijzing aan de deelverzameling $(v_h|h\in Q_j)$ overeenkomstig de partitie τ_j zodanig dat elk element van deze deelverzameling alleen afhankelijk is van de elementen van de deelverzameling $(y_h|h\in Q_j \cap j\in P_i)$

Ter illustratie van de stelling 3.2.B zie machine C (fig.4)

	00	01	11
1	6	1	3
2	5	2	4
3	4	1	5
4	3	2	6
5	2	3	5
6	1	4	6

	v_1	v_2	v_3
1	0	0	0
2	0	0	1
3	1	0	0
4	1	0	1
5	0	1	0
6	0	1	1

fig.4. Machine C en een toewijzing.

Twee bijbehorende partitieparen zijn:

$(\overline{1,2};\overline{3,4};\overline{5,6})\,(\overline{1,2};\overline{3,4};\overline{5,6})=(\pi_1,\tau_1)$
$(1,3,5;2,4,6)\,(1,3,5;2,4,6)=(\pi_2,\tau_2)$

Uit stelling 3.2.B volgt voor i=1 dat $P_i=(1)$.

Dit betekent dat de variabelen,gebruikt voor de toewijzing met behulp van τ_1 van zichzelf afhankelijk zijn. (en natuurlijk ook van de input).

τ_1 geeft aanleiding tot het toekennen aan de variabelen Y_1 en Y_2. Er volgt dan:

$Q_i=Q_1=(1,2)$ en $(v_h|\,h\in Q_1\cap j\in P_1)=(v_1,v_2)$.

Dan is het resultaat: $Y_1=f(v_1,v_2,x_1,x_2)$ en

$$Y_2=f(v_1,v_2,x_1,x_2).$$

Voor i=2 volgv $P_i=(2)$.

τ_3 geeft aanleiding tot het toekennen van de variabele v_3; dan is $Q_2=(3)$ dus $(v_h|\,h\in Q_2\cap j\in P_2)=(v_3)$.

$Y_3$is dus van zichzelf en van de input afhankelijk.

$$Y_3=f(v_3,x_1,x_2).$$

Opstellen van de logische vergelijkingen levert inderdaad de afhankelijkheid op zoals hierboven reeds is afgeleid.

$Y_1=\bar{v}_1\bar{v}_2x_1+v_1\bar{x}_2+v_2\bar{x}_1x_2$

$Y_2=\bar{v}_1\bar{v}_2\bar{x}_2+v_1x_1+v_2x_1$

$Y_3=\bar{v}_1\bar{x}_2+v_1x_2$

In de praktijk zullen voor de toestandstoekenning bij een machine M vaak tweeblokspartities worden gebruikt. Om een geschikte keuze te maken uit een aantal rechter partities van alle partitieparen worden eerst alle Mm-paren gegenereerd.

Van een Mm-paar mag volgens lemma 3.2 de rechter partitie altijd vergroot worden tot een tweeblokspartitie; het nieuwe stel partities blijft een partitiepaar.

Uit de op deze wijze ontstane partitieparen wordt nu een keuze gemaakt voor toekenningen waarbij stelling

3.2.B een leidraad vormt voor de keuze.
Men dient er echter wel op te letten dat na de toekenningen de verschillende toestanden van de machine te onderscheiden zijn.

lemma 3.3.
Bij een toekenning van de toestanden van een machine M aan de hand van een aantal partities τ_j (j=1 tm k) moet voldaan zijn aan:

$$\prod(\tau_j) = 0$$

om de verschillende toestanden na toekenning van elkaar te onderscheiden.

Bewijs:
We doen een toekenning met behulp van τ_1.
De toestanden in een blok van τ_1 zijn niet van elkaar te onderscheiden.
Na een tweede toekenning met behulp van τ_2 zijn de toestanden die in de blokken van $(\tau_1 . \tau_2)$ zitten niet van elkaar te onderscheiden.
Dit proces zetten we voort totdat alle blokken slechts een toestand hebben; dan is $\prod(\tau_j)=0$ en alle toestanden zijn van elkaar te onderscheiden. Q.E.D.

We zullen tenslotte nog een voorbeeld behandelen van een machine D.(fig. 5)
Van deze machine zijn de Mm-paren:

$(\pi_1,\tau_1) = ((\overline{1,2,4};\overline{3,5}),(\overline{1,3,5};\overline{2};\overline{4}))$
$(\pi_2,\tau_2) = ((\overline{1,3};\overline{2};\overline{4,5}),(\overline{1,3,4};\overline{2,5}))$
$(\pi_3,\tau_3) = ((\overline{1,4};\overline{3,5};\overline{2}),(\overline{1};\overline{2};\overline{3,5};\overline{4}))$
$(\pi_4,\tau_4) = ((\overline{1};\overline{2,4};\overline{3};\overline{5}),(\overline{1,5};\overline{2};\overline{3};\overline{4}))$ en de triviale Mm-paren

(I,I) en (0,0).

We vergroten nu de partities τ_i tot tweebloksparttities τ_i' en zoeken naar een aantal τ_i' zodanig dat :

1) $\Pi(\tau_i')=0$ en 2) er zoveel mogelijk verminderde afhankelijkheid optreedt van de secundaire variabelen.

Na enig proberen met toepassing van stelling 3.2.B vinden we de volgende partitieparen.

$(\pi_1,\tau_1')=((\overline{1,2,4};\overline{3,5}),(\overline{1,3,4,5};\overline{2}))$

$(\pi_3,\tau_3')=((\overline{1,4};\overline{3,5};\overline{2}),(\overline{1,2,4};\overline{3,5}))$

$(\pi_4,\tau_4')=((\overline{1};\overline{2,4};\overline{3};\overline{5}),(\overline{1,5};\overline{2,3,4}))$

Er volgt nu door met behulp van τ_i' de toestanden aan Y_i toe te kennen:

$\pi_1 \geq \tau_3'$ dus $Y_1=f(v_3,x_1,x_2)$

$\pi_3 \geq \tau_1'.\tau_3'$ dus $Y_3=f(v_1,v_3,x_1,x_2)$

$\pi_4 \geq \tau_3'.\tau_4'$ dus $Y_4=f(v_3,v_4,x_1,x_2)$

en $(\tau_1'.\tau_3'.\tau_4')=0$.

De toekenningstabel ziet er dan als volgt uit:

	v_1	v_2	v_3
1	0	0	0
2	1	1	0
3	0	1	1
4	0	1	0
5	0	0	1

De logische vergelijkingen worden dan:

$Y_1= x_1\overline{x}_2\overline{v}_3$

$Y_3= x_1v_3+\overline{x}_1\overline{x}_2\overline{v}_1+\overline{x}_1x_2v_1$

$Y_4= x_1\overline{v}_3+x_2v_3+\overline{x}_1v_4v_3+\overline{x}_2\overline{v}_4\overline{v}_3$

3.1. Definities en eigenschappen t.a.v. verlengde partities bij machines met don't care condities.

Ook bij don't care condities kan met succes gebruik worden gemaakt van de partitie-paren-theorie uit hoofdstuk 3 zij het in gewijzigde vorm.

Machine E(fig. 6a) is een machine met d.c. condities.

	a	b	c
1	4	1	3
2	-	2	4
3	1	3	4
4	2	3	-

fig. 6a. Machine E met d.c. condities

	a	b	c
1	4	1	3
2	5	2	4
3	1	3	4
4	2	3	6

fig. 6b. Machine E met gespecificeerde d.c. condities

Door nu de d.c. condities nader te specificeren is het mogelijk een definitie te geven van een verlengd partitie-paar.

Noem de verzameling van oorspronkelijke toestanden S_1 en de verzameling van toestanden na de specificatie S_r.

In het bovenstaande voorbeeld is:

$S_1=(1,2,3,4)$ en $S_r=(1,2,3,4,5,6)$.

<u>definitie 3.4</u>.

Gegeven een machine $M=(S_1 I,O,\delta,\lambda)$ met gespecificeerde d.c. condities en een partitie π op S_1 en τ op S_r.

Dan wordt (π,τ) een verlengd partitiepaar genoemd dan en alleen dan als :

$s_i \equiv s_j(\pi)$ inhoudt dat $\delta(s_i,x)\equiv\delta(s_j,x)(\tau)$, voor elke $x\in I$.

Voor machine E zijn:

$$(\pi_1,\tau_1)=((\overline{1};\overline{2,4};\overline{3}),(\overline{1};\overline{2,3,5};\overline{4,6}))\text{ en}$$

$$(\pi_2,\tau_2)=((\overline{1};\overline{2,3};\overline{4}),(\overline{1,5};\overline{2,3};\overline{4};\overline{6}))\text{ en}$$

$$(\pi_3,\tau_3)=((\overline{1,2};\overline{3,4}),(\overline{1,2};\overline{3,4,5,6}))$$

verlengde partitieparen.

Alle eigenschappen van gewone partities en partitieparen zijn ook van toepassing voor verlengde partitieparen met uitzondering van lemma 3.3 en de hoofdstelling 3.2. Door nu eerst een restrictie $\overline{\tau}$ van de verlengde partitie τ in te voeren kunnen stelling 3.2.A en 3.2.B en lemma 3.3 letterlijk worden overgenomen indien voor τ_j dan $\overline{\tau_j}$ en voor $\tau(y_j)$, $\overline{\tau}(y_j)$ wordt ingevuld.

$\overline{\tau_j}$ moet dan voldoen aan de volgende eigenschap:

$s_a \equiv s_b\,(\overline{\tau_j})$ met s_a en s_b in S_1 dan en alleen dan als $s_a \equiv s_b\,(\tau_j)$

Voorbeeld:

De restricties van bovenstaande verlengde partities τ_1,τ_2 en τ_3 zijn dan:

$$\overline{\tau_1}=(\overline{1};\overline{2,3};\overline{4})$$

$$\overline{\tau_2}=(\overline{1};\overline{2,3};\overline{4});\ \overline{\tau_2'}=(\overline{1,4};\overline{2,3})$$

$$\overline{\tau_3}=(\overline{1,2};\overline{3,4})$$

Met behulp van bijv. τ_2' en τ_3 kunnen we toewijzingen doen daar $\overline{\tau_2'}.\overline{\tau_3}=0$.

Omdat $\overline{\tau_3}\leq\pi_3$ is y_3 slechts afhankelijk van zichzelf en de input en aangezien $\overline{\tau_3}.\overline{\tau_2}\leq\pi_2$ is y_2 afhankelijk van zichzelf, van y_3 en van de input.

De toewijzing is dan overeenkomstig de onderstaande tabel:

	1	2	3	4	5	6
y_2	0	1	1	0	0	0
y_3	0	0	1	1	1	1

Met a=00, b=01 en c=11 volgt voor machine E:

$Y_2=\overline{y}_2\overline{y}_3x_1+y_2\overline{x}_1x_2+\overline{y}_2y_3\overline{x}_1$; $Y_3=\overline{y}_3\overline{x}_2+\overline{y}_3x_1+y_3\overline{x}_1x_2$

3.2. Definities en eigenschappen t.a.v. I-S partitieparen.

Het blijkt in vele gevallen nuttig te zijn om naast het tot nu toe gedefiniëerde partitiepaar (π,τ), waarbij de linker en de rechter partitie betrekking hadden op de toestanden (S-S paar), nog een partitiepaar te definiëren waarbij de linker partitie betrekking heeft op de verzameling van inputs $(x_1 \dots x_n)$ en de rechter partitie weer op de verzameling van toestanden. Dit partitiepaar heet een I-S paar.

Definitie 3.5.

Van een machine $M=(S,I,O,\delta,\lambda)$ is (ξ,τ) een I-S paar indien de partitie ξ betrekking heeft op de inputs $I=(x_1 \dots x_n)$ en de partitie τ op de toestanden $S=(s_1 \dots s_m)$, als voor alle s_i geldt dat voor $x_u \equiv x_v (\xi)$ volgt dat $\delta(s_i,x_u) \equiv \delta(s_i,x_v)(\tau)$.

Voorbeeld: Machine E (fig.6b)

Voor deze machine zijn bijv.

$(\xi_1,\tau_1)=((\overline{a,b};\overline{c}),(\overline{1,2,3,4,5};\overline{6}))$

$(\xi_2,\tau_2)=((\overline{a};\overline{b,c}),(\overline{1,2,3,4,6};\overline{5}))$ en

$(\xi_3,\tau_3)=((\overline{a,c};\overline{b}),(\overline{1,3,4,5};\overline{2,6}))$ I-S partitieparen.

Definities, stellingen, en algorithmen met betrekking tot I-S paren of Mm-I-S paren zijn volkomen analoog aan die van S-S paren met dien verstande dat de linker partitie betrekking heeft op de Input, en indien van verminderde afhankelijkheid wordt gesproken, dan heeft dit ook betrekking op de input.

4. Het algorithme voor de berekening van de Mm-paren.

Door Hartmanis en Stearns [1] is een algorithme beschreven voor het maken van Mm-paren van een machine M.
Bij dit algorithme wordt uitgegaan van de kleinst mogelijke partitieparen $(\pi_{st}, m(\pi_{st}))$.
Door vergelijken en optellen van de rechter en linker partities worden alle Mm-paren gemaakt.
Het hier beschreven algorithme is iets afwijkend van het algorithme van Hartmanis en Stearns.
Met behulp van de $(\pi_{st}, m(\pi_{st}))$ partitieparen worden eerst basis Mm-paren gemaakt.
Alle basis Mm-paren vormen de eerste generatie.
Door onderlinge combinaties van de basis Mm-paren wordt een 2^e generatie Mm-paren gegenereerd.
Deze generatie wordt weer gecombineerd met de 1^e en levert dan de 3^e generatie op.
Dit proces wordt voortgezet totdat geen nieuwe Mm-paren meer gegenereerd kunnen worden.
Het algorithme verloopt als volgt:

1) Van een machine M worden alle linker partities gemaakt die slechts in een blok twee toestanden (s en t) onderbrengen en in alle andere blokken een toestand.
Noem deze partities π_{st}.

2) Uit de volgordetabel wordt $m(\pi_{st})$ bepaald voor elke π_{st}.
Dit gaat als volgt:
s en t komen in een blok voor; dan komen ook $\delta(s,x)$ en $\delta(t,x)$ in een blok voor.

Voorbeeld: machine F (fig. 6.b)

$(\pi_{12}, m(\pi_{12})) = ((\overline{1,2};\overline{3,4}), (\overline{1,2};\overline{3,4,5};\overline{6}))$

Deze partitie beschouwen we voorlopig als basis "Mm-paar".

Zijn er $(\pi_{st}, m(\pi_{st}))$-paren z.d.d. $m(\pi_{st}) \leq m(\pi_{12})$ dan wordt het nieuwe basis "Mm-paar" : $(\pi_{12}, m(\pi_{12})) := (\pi_{12}+\pi_{st}, m(\pi_{12}))$.

$(\pi_{13}, m(\pi_{13})) = ((\overline{1,3};\overline{2};\overline{4}), (\overline{1,3,4};\overline{2};\overline{5};\overline{6}))$.

$m(\pi_{13})$ is ongelijk aan een eerder gevonden basis"Mm-paar" en het paar wordt dan verder beschouwd als een basis "Mm-paar".

Zijn er rechtse partities in de lijst van basis Mm-paren die kleiner zijn als $m(\pi_{13})$ dan wordt het nieuwe basis "Mm-paar $(\pi_{13}, m(\pi_{13})) := (\pi_{13}+\pi_{st}, m(\pi_{13}))$.

Dit is hier echter niet het geval. Dus het oorspronkelijke als zodanig aangenomen basis Mm-paar blijft hetzelfde.

$(\pi_{14}, m(\pi_{14})) = ((\overline{1,4};\overline{2};\overline{3}), (\overline{1,3,6};\overline{2,4};\overline{5}))$

$(\pi_{23}, m(\pi_{23})) = ((\overline{1};\overline{2,3};\overline{4}), (\overline{1,5};\overline{2,3};\overline{4};\overline{6}))$

$(\pi_{24}, m(\pi_{24})) = ((\overline{1};\overline{2,4};\overline{3}), (\overline{1};\overline{2,3,5};\overline{4,6}))$

$(\pi_{34}, m(\pi_{34})) = ((\overline{1};\overline{2};\overline{3,4}), (\overline{1,2};\overline{3};\overline{4,6};\overline{5}))$

Na de correctie van alle basis "Mm-paren" met $(\pi_{34}, m(\pi_{34}))$ en met alle basis "Mm-paren op $(\pi_{34}, m(\pi_{34}))$ hebben we nu de echte basis Mm-paren overgehouden.

Hadden we een basis Mm-paar gevonden waarvan $m(\pi_{st})$ gelijk was aan een $m(\pi_{ij})$ uit de lijst van basis "Mm-paren" dan voeren we met $(\pi_{st}, m(\pi_{st}))$ wel correcties uit op de basis "Mm-paren". We voegen $(\pi_{st}, m(\pi_{st}))$ echter niet toe aan de lijst.

3) Nu moeten de volgende generaties Mmparen worden gegenereerd

Het verdere verloop van het algorithme is gebaseerd op de volgende stellingen.

(Noem een basispartitie π_B en het Mm-paar (π_B, τ_B))

Stelling 4.1.

Indien (π, τ) een Mm-paar is dan is

$$\tau = \sum (m(\pi_{st}) | \pi_{st} \leq \pi)$$

Het bewijs wordt geleverd door Hartmanis en Stearns[1]. Uit lemma 3.1 volgt dan dat $\pi = \sum (\pi_{st} | \pi_{st} \leq \pi)$. Een gevolg van stelling 4.1 is nu dat elke partitie dan ook te schrijven is als : $\pi = \sum (\pi_B | \pi_B \leq \pi)$. daar elke π gevormd wordt door de som van de elementen van een deelverzameling (π_{st}). Indien in deze deelverzameling elementen simultaan optreden dan vormt de som van deze simultaan optredende elementen een π_B

Stelling 4.2.

Indien (π, τ) een Mm-paar is dan is

$$\tau = \sum (m(\pi_B) | \pi_B \leq \pi).$$

Bewijs:

Daar $\pi \geq \pi_B$ is (π_B, τ) een partitiepaar (lemma 3.2), en daarom is $m(\pi_B) \leq \tau$.

Er volgt nu: $\sum (m(\pi_B) | \pi_B \leq \pi) \leq \tau$ A)

daar de som van partities, die allen kleiner zijn dan een partitie τ, nooit groter kan worden als τ.

Daar $\pi = \sum (\pi_B | \pi_B \leq \pi)$ en $\sum (m(\pi_{st}) | \pi_{st} \leq \pi) = \sum (m(\pi_B) | \pi_B \leq \pi)$

volgt: $\left(\sum (\pi_B | \pi_B \leq \pi), \sum (m(\pi_B) | \pi_B \leq \pi)\right) = \left(\pi, \sum (m(\pi_B) | \pi_B \leq \pi)\right)$

is een partitiepaar.

Dan is $\sum (m(\pi_B) | \pi_B \leq \pi) \geq m(\pi) = \tau$ B)

Uit A) en B) volgt dan dat:

$$\sum (m(\pi_B) | \pi_B \leq \pi) = \tau \qquad \text{Q.E.D.}$$

Een belangrijke conclusie van stelling 4.2 is dat van alle Mm-paren de rechter partitie de som is van de rechter basis partities.
We zijn nu in staat om met de tot nu toe verkregen basispartities door optellen de volgende generaties te maken.

Voor machine E wordt de 2^e generatie als volgt gegenereerd:
τ^{21}(2^e generatie, 1^e paar) $= (\tau_{B1}+\tau_{B2}) = (\overline{1,2,3,4,5};\overline{6})$.
We gaan nu na welke rechter partities van de basis Mm-paren kleiner of gelijk zijn aan τ^{21}.
Vinden we een gelijkheid dan hebben we het betreffende partitiepaar al eerder gevonden.
We geven π^{21} de waarde 0.
Alle rechter partities $(\pi_{Bi}|\tau_{Bi}<\tau^{21})$ worden opgeteld bij π^{21}.
Zijn we de hele lijst basis Mm-paren afgelopen dan heeft π^{21} zijn eindwaarde gekregen en vormt (π^{21},τ^{21}) een Mm-paar van de 2^e generatie.
Voor bovenstaand geval wordt:
$\pi^{21}=\pi_{B1}+\pi_{B2}+\pi_{B4}=(\overline{1,2,3};\overline{4})$ dus $(\pi^{21},\tau^{21})=((\overline{1,2,3};\overline{4}),(\overline{1,2,3,4,5};\overline{6}))$
τ^{22} wordt: $(\tau_{B1}+\tau_{B3})=(\overline{1,2,3,4,5,6})=I$.
Dit is een triviale rechter partitie en kan vergeten worden.
$\tau^{22}=(\tau_{B4}+\tau_{B1})=(\overline{1,2,3,4,5};\overline{6})$.
Deze partitie wordt eerst getest op gelijkheid met een partitie uit de 2^e generatielijst.
Vinden we geen gelijke dan wordt getest en opgeteld met behulp van de basis Mm-paren lijst.
Voor τ^{22} vinden we een gelijkheid in de 2^e generatie-lijst: $\tau^{22}=\tau^{21}$, dus het betreffende paar hebben we al.

$\tau^{22} = \tau_{B1} + \tau_{B6} = I$ dus vervalt.

$\tau^{22} = \tau_{B1} + \tau_{B6} = (\overline{1,2};\overline{3,4,5,6})$.

Deze staat nog niet in de 2^e generatielijst.

τ_{B1} en τ_{B6} zijn de enigste partities die kleiner zijn dan τ^{22} dus π^{22} wordt $\pi_{B1} + \pi_{B6} = (1,2;3,4)$ dus $(\pi^{22},\tau^{22}) = ((\overline{1,2};\overline{3,4}),(\overline{1,2};\overline{3,4,5,6}))$.

De procedure wordt voortgezet totdat alle mogelijke 2-combinaties van de τ_{Bi} zijn gemaakt.

Combinatie van de 2^e generatie met de 1^e levert geen nieuwe 3^e generatie partities op daar alle τ^{2i} tweebloks-partities zijn.

Voor machine F zijn dan alle Mm-paren berekend.

Aan de hand van machine F(fig. 7) wordt het hele algorithme nog eens doorlopen.

Omdat in de 1^e generatielijst de linkse partities tijdens het verloop van de berekening kunnen veranderen, zal terwille van de vereenvoudigde notatie een p.p. (π,τ) omgekeerd worden genoteerd (τ,π).

Heeft π de waarde $\pi_1, \pi_2, \pi_3 \ldots \pi_n$ voordat het paar (π,τ) een Mm-paar is, dan wordt dit als volgt genoteerd:

$(\tau)\,(\cancel{\pi_1})\,(\cancel{\pi_2})\,.\,(\cancel{\pi_n})\ldots(\pi)$.

	x_1	x_2
1	2	3
2	4	5
3	5	6
4	1	1
5	1	1
6	1	1

Fig. 7. Machine F.

Het verloop van de berekening gaat dan als volgt:

<u>1^e generatie</u>

$(\overline{1};\overline{2,4};\overline{3,5};\overline{6})(\overline{1,2};\overline{3};\overline{4};\overline{5};\overline{6}) = (\tau_1,\pi_1)$

$(\overline{1};\overline{2,5};\overline{3,6};\overline{4})(\overline{1,3};\overline{2};\overline{4};\overline{5};\overline{6}) = (\tau_2,\pi_2)$

$(\overline{1,2,3};\overline{4};\overline{5};\overline{6})\cancel{(\overline{1,4};\overline{2};\overline{3};\overline{5};\overline{6})}\cancel{(\overline{1,4,5};\overline{2};\overline{3};\overline{6})}(\overline{1,4,5,6};\overline{2};\overline{3}) = (\tau_3,\pi_3)$

$(\overline{1};\overline{2};\overline{3};\overline{4,5,6})(\overline{1};\overline{2,3};\overline{4};\overline{5};\overline{6}) = (\tau_4,\pi_4)$

$(\overline{1,4,5};\overline{2};\overline{3};\overline{6})\cancel{(\overline{1};\overline{2,4};\overline{3};\overline{5};\overline{6})}\cancel{(\overline{1};\overline{2,4,5};\overline{3};\overline{6})}(\overline{1};\overline{2,4,5,6};\overline{3}) = (\tau_5,\pi_5)$

$(\overline{1,5,6};\overline{2};\overline{3};\overline{4})\cancel{(\overline{1};\overline{2};\overline{3,4};\overline{5};\overline{6})}\cancel{(\overline{1};\overline{2};\overline{3,4,5};\overline{6})}(\overline{1};\overline{2};\overline{3,4,5,6}) = (\tau_6,\pi_6)$

$(\overline{1};\overline{2};\overline{3};\overline{4};\overline{5};\overline{6})\cancel{(\overline{1};\overline{2};\overline{3};\overline{4,5};\overline{6})}(\overline{1};\overline{2};\overline{3};\overline{4,5,6}) = (\tau_7,\pi_7)$

Combinatie van de toestanden 5 en 6 levert geen nieuwe bijdrage meer; de eerste generatielijst is dan klaar.

<u>2^e generatie</u>

$(\overline{1};\overline{2,3,4,5,6})(\overline{1,2,3};\overline{4,5,6}) = (\tau_1+\tau_2,\pi_1+\pi_2+\pi_7)$

$(\overline{1,2,3,4,5};\overline{6})(\overline{1,2,4,5,6};\overline{3}) = (\tau_1+\tau_3,\pi_1+\pi_3+\pi_5+\pi_7)$

$(\overline{1,3,5,6};\overline{2,4})(\overline{1,2};\overline{3,4,5,6}) = (\tau_1+\tau_6,\pi_1+\pi_6+\pi_7)$

$(\overline{1,2,3,5,6};\overline{4})(\overline{1,3,4,5,6};\overline{2}) = (\tau_2+\tau_3,\pi_2+\pi_3+\pi_6+\pi_7)$

$(\overline{1,2,4,5};\overline{3,6})(\overline{1,3};\overline{2,4,5,6}) = (\tau_2+\tau_5,\pi_2+\pi_5+\pi_7)$

$(\overline{1,2,3};\overline{4,5,6})(\overline{1,4,5,6};\overline{2,3}) = (\tau_3+\tau_4,\pi_3+\pi_4+\pi_7)$

$(\overline{1,4,5,6};\overline{2};\overline{3})(\overline{1};\overline{2,3,4,5,6}) = (\tau_4+\tau_5,\pi_4+\pi_5+\pi_6+\pi_7)$

Alle mogelijke combinaties van de 1^e generatielijst leveren geen nieuwe partitieparen meer op.
Combinatie van de 1^e generatielijst met de 2^e levert partitieparen op die we al hadden of triviale partitieparen (π,I).
In de 1^e en tweede generatielijst zijn dus alle Mm-paren aanwezig.

4.1. Conclusie.

In de appendix vindt U het geprogrammeerde algorithme met de berekende Mm-paren van een aantal machines.
Het algorithme werd geprogrammeerd in PL/1.
De benodigde vertaaltijd is ongeveer 10 min.
Een belangrijk nadeel van het algorithme is de benodigde rekentijd, veroorzaakt door de noodzakelijke vergelijkingen van rechter partities nadat men een nieuwe partitie heeft gevonden. Vooral indien men veel basis Mm-paren heeft gevonden waarvan de rechter partities klein zijn (veel blokken), dan zijn er veel combinatiemogelijkheden die allemaal weer gecontroleerd moeten worden.
Programmatechnisch zijn er nog aanzienlijke verbeteringen aan te brengen die de rekentijd een flink stuk kunnen beperken.
De rekenmachine zet een partitie in het geheugen door het bloknummer van een toestand vast te leggen.
Het bloknummer is willekeurig; bijvoorbeeld:

blokno. : 2 1 3

partitie: $=(\overline{1,2,5,7};\overline{3,8};\overline{4,6})$

Dan is de notatie in het geheugen:

toestand:	1	2	3	4	5	6	7	8
blokno. :	2	2	1	3	2	3	2	1

Om na te gaan of tussen twee partities de relatie $<,=,>$ of $\neq$ bestaat wordt een procedure doorlopen.
Vooral de controle op gelijkheid komt vaak voor.
(Alle nieuwe rechter partities moeten gecontroleerd worden op gelijkheid met alle reeds gevonden partities.)
Indien nu een meer uniforme manier van notatie in het geheugen was gebruikt, dan had men i.p.v. de procedure

voor deze steeds terugkomende controle op gelijkheid slechts een if-statement hoeven te gebruiken.
Een eenduidige en uniforme manier van notatie zou bijvoorbeeld te verkrijgen zijn door de blokken te nummeren in de volgorde die aangegeven wordt door de toestand met het laagste nummer in het blok.
Voorbeeld:

blokno. : 1 2 3

partitie: $=(\overline{1,2,5,7};\overline{3,8};\overline{4,6})$

De notatie is dan:

toestand:	1	2	3	4	5	6	7	8
blokno. :	1	1	2	3	1	3	1	2

Het is van te voren niet te voorspellen hoeveel Mm-paren bij een bepaalde machine horen.
Naarmate het aantal toestanden groter wordt zal gemiddeld het aantal Mm-paren ook groter worden.
Voor het aantal inputs is juist het omgekeerde het geval; naarmate het aantal inputs kleiner zal zijn zullen gemiddeld de rechter partities van de basis Mm-paren ook kleiner zijn. Algemeen zijn er dus meer combinatiemogelijkheden te vinden omdat er meer partities nodig zijn die als som de Eenheidspartitie opleveren.

Het geprogrammeerde algorithme kan maximaal 200 partitie-paren berekenen.
Het aantal te verwerken toestanden is maximaal 16.
Hierbij zijn de gespecificeerde don't cares meegeteld.
Ook het aantal maximaal te verwerken inputs is 16.
Verhoging van het aantal toestanden is in principe

wel mogelijk indien men de partitieparen niet meer in het kerngeheugen van de rekenmachine wil schrijven maar bijvoorbeeld op de disk.
Er moet dan wel rekening gehouden worden met een aanzienlijk hogere rekentijd.

Indien men vasthoud aan het criterium van de verminderde afhankelijkheid bij het toekennen van de toestanden aan de secundaire variabelen dan zal een gunstige selectie uit de Mm-paren lijst over het algemeen erg moeilijk worden, omdat in vele gevallen deze lijst nog altijd een behoorlijke keuzemogelijkheid aan de ontwerper overlaat.

1) Hij kiest een aantal rechter partities (vergroot tot tweebloksparitities) die een doorsnede 0 hebben.
Indien een partitie uit n blokken bestaat dan kunnen op $(2^{n-1}-1)$ verschillende manieren tweebloksparitities worden gemaakt.
2) De partities moeten getest worden aan de hand van het criterium van verminderde afhankelijkheid.
3) Uit alle mogelijkheden moet de beste gekozen worden.

Indien er voor een grotere machine geen voor de hand liggende oplossing bestaat, is het vaak een immense taak een behoorlijk optimale selektie te krijgen.

Het is bovendien in het algemeen de vraag of het criterium van verminderde afhankelijkheid wel de beste oplossing geeft.

In sommige gevallen zal men door proberen en redeneren tot een goedkopere versie van een machine kunnen komen. Hiervoor bestaat echter nog geen algemene aanpak,zodat voorlopig het kriterium van verminderde afhankelijkheid, met in alle gevallen tenminste een gunstig eindresultaat, gehanteerd kan worden.

5. Toekenningen met behulp van tweeblokspartities.

5.1. Inleiding.

Het is vaak nog een hele toer om met behulp van Mm-paren een geschikte toewijzing te doen.

Indien gebruik gemaakt wordt van binaire geheugenelementen (flipflops) dan zal een toewijzing aan de secundaire variabelen geschieden aan de hand van tweeblokspartities τ.

Er bestaat nu een betrekkelijk eenvoudige manier om de $M(\tau)$-partitie te bepalen.

Met behulp van de stelling die de afhankelijkheid van de secundaire variabelen $Y(\tau)$ aangeeft, kan men dan weer voor een bepaalde machine toewijzingen doen voor een optimale verminderde afhankelijkheid.

Een hulpmiddel voor het berekenen van $M_{s-s}(\tau)$ en $M_{i-s}(\tau)$ zijn resp. de rij- en kolomnummers.

5.2 Het genereren van rij- en kolomnummers.

Stel dat we een toewijzing doen aan de hand van een tweeblokspartitie τ met $\tau=(\overline{1,2,4,6.};\overline{3,5})$.

Alle toestanden in het eerste blok van τ krijgen een "0" toegewezen en alle toestanden in het tweede blok een "1".

Voor dit geval:

toestandsnummer:	1,2,3,4,5,6
toewijzing:	0,0,1,0,1,0

Elke toewijzing is dus te noteren als een binair getal.

Verder bepaalt een toewijzing eenduidig een tweeblokspartitie.

We noteren nu niet meer de toestanden die in een bepaald blok zitten, doch voor elke toestand noteren we in welk blok deze voorkomt.

De verschillende blokken worden op een verschillende manier genummerd. B.v. $\tau = (001010)$ betekent dat de toestanden 1,2,4 en 6 in een blok (aangeduid met een "1") voorkomen.

Algemeen:

Oude schrijfwijze:

$\pi = (\overline{1,2,4};\overline{3,6,9};\overline{5,7};\dot{\overline{8}})$. We nummeren nu de blokken van 1 tm. 4. De nieuwe schrijfwijze wordt dan:

$\pi = (112132342)$.

Een tweeblokspartitie en de bijbehorende toewijzing kunnen we dus weergeven als een binair getal dat we op papier in het decimale stelsel, of nog handiger in het achttallige stelsel kunnen noteren.

We zullen ons in het vervolg beperken tot het achttallige stelsel; dus $\tau = (001010) = (12)$.

Om een uitspraak te kunnen doen over de afhankelijkheid van een bepaalde secundaire variabele $Y(\tau)$ met de $M(\tau)$ worden bepaald.

Hiertoe wordt een bepaalde toekenning in de volgordetabel voor de toestanden gesubstitueerd.

In fig. 8a en 8b is dit voor een tweetal toekenningen gedaan aan de hand van de verlengde tweeblokspartities (01) en (30).

	a	b	c	S01
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	1	1
I01	00	00	01	

fig. 8a

toekenning(01)voor

machine E.

	a	b	c	S30
1	0	0	1	1
2	0	1	0	2
3	0	1	0	2
4	1	1	0	6
I30	01	07	10	

fig. 8b

toekenning(30)voor

machine E.

De achttallige weergave van het binaire getal dat in een rij van de tabel staat wordt genoteerd achter deze rij en wordt een rijnummer genoemd:

notatie: S01=(0,0,0,1);

S30=(1,2,2,6).

De achttallige weergave van het binaire getal dat in een kolom van de tabel staat wordt genoteerd onder deze kolom en wordt kolomnummer genoemd;

notatie: I01=(00,00,01);

I30=(01,07,10).

De volgende stelling zal het praktische nut van rij- en kolomnummers doen inzien:

Stelling 5.1.a:

Indien voor twee rijnummers Sn(s) en Sn(t), behorende bij de toestandstoekenning(n) geldt dat Sn(s)=Sn(t), dan geldt voor $\tau=\tau(n)$ dat:

$$s \equiv t[M_{s-s}(\tau)].$$

Stelling 5.1.b:

Indien voor twee kolomnummers Im(i) en Im(j) behorende bij de toestandstoekenning (m) geldt dat Im(i)=Im(j), dan geldt voor $\tau=\tau(m)$ dat:

$$i\equiv j\,[M_{i-s}(\tau)].$$

Bewijs:

We zullen 5.1.a bewijzen, het bewijs voor 5.1.b is identiek.

Als de rijnummers hetzelfde zijn dan betekent dat indien:

$\delta(s,x)=u$, met u 0 of u 1;

$\delta(t,x)=v$, met v 0 of v 1;

dat voor $u\to 1$ ook geldt:$v\to 1$ en

$u\to 0$ geldt:$v\to 0$ voor alle mogelijke inputs $x\in I$.

Hieruit volgt dat a en b in hetzelfde blok zitten van de partitie .Dus:

$$\delta(s,x)\equiv\delta(t,x)\,[\tau] \qquad \text{C)}$$

Beschouw nu de partitie π waarbij s en t in hetzelfde blok voorkomen. Volgens C) is (π,τ) dan een partitiepaar.

We moeten nu nog bewijzen dat $\pi=M_{s-s}(\tau)$.

Veronderstel $\pi_g>\pi$ en:

$u\equiv v(\pi_g)$ D)

$u\not\equiv v(\pi)$ E)

$\delta(u,x)=p$ F)

$\delta(v,x)=q$ G)

dan volgt uit E) dat:

$(p,q)\to(0,1)$ of $(p,q)\to(1,0)$ H)

Indien (π_g,τ) een partitiepaar zou zijn dan had moeten gelden dat p en q in hetzelfde blok van τ hadden moeten voorkomen.

Dit is strijdig met H), dus er bestaat geen partitiepaar (π_g,τ) met $\pi_g>\pi$.

Hieruit volgt dan $\pi=M(\tau)$. QED.

Om alle rijnummers te genereren is het niet nodig alle toekenningen in de tabel te substitueren.

Eenvoudig te verifiëren is:

$Sn(\text{achttal})=Sn(\text{binair})=S0b_2b_3\ldots\ldots b_m$

met $0b_2b_3\ldots\ldots b_m$ is een binair getal; b_i="0" of "1"; en m is het totale aantal toestanden dat betrekking heeft op de machine.

$S0b_2b_3\ldots b_m=S0b_20\ldots0+S00b_3\ldots0+\ldots\ldots+S000\ldots b_m$.

Nodig is dus alleen maar de toekenningen te substitueren waarbij achtereenvolgens de toestanden 2,3,...m een "1" krijgen toegewezen en de rest een "0".

Uit de tabel volgen dan de rij- en de kolomnummers voor deze toekenning. Alle overige toekenningen kunnen door optellen gevonden worden.

Voorbeeld:

S26=S010110=S010000+S000100+S000010=S20+S04+S02

Voor machine E(fig 6b) kunnen alle rij- en kolomnummers gegenereerd worden uit de rij- en de kolomnummers behorende bij de toekenningen (01),(02),(04),(10) en (20).

In de tabel van fig. 9 staan de rij- en kolomnummers behorende bij alle mogelijke toekenningen.

De notatie voor rij- en kolomnummers komt overeen met de nieuwe notatie voor partities, alleen de blokken zijn op een willekeurige manier genummerd.

5.3. Het profiel.

In het algemeen zal men uit de tweeblokspartities een keuze willen maken waarbij een zo groot mogelijke ver= minderde afhankelijkheid tav de toestandsvariabele Y en

tweebloks partitie	rijnummers 1	2	3	4	aantal var.	kolomnummers a	b	c	aantal var.
* 01	0	0	0	1		00	00	01	
* 02	0	4	0	0		04	00	00	
* 03	0	4	0	1		04	00	01	
* 04	4	1	1	0		10	00	06	
* 05	4	1	1	1		10	00	07	
* 06	4	5	1	0		14	00	06	
* 07	4	5	1	1		14	00	07	
* 10	1	0	2	2		00	03	10	
* 11	1	0	2	3		00	03	11	
* 12	1	4	2	2		04	03	10	
* 13	1	4	2	3		04	03	11	
14	5	1	3	2	2	10	03	16	2
15	5	1	3	3	2	10	03	17	2
16	5	5	3	2	2	14	03	16	2
17	5	5	3	3	1	14	03	17	2
* 20	0	2	0	4		01	04	00	
* 21	0	2	0	5		01	04	01	
* 22	0	6	0	4		05	04	00	
* 23	0	6	0	5		05	04	01	
24	4	3	1	4	2	11	04	06	2
25	4	3	1	5	2	11	04	07	2
26	4	7	1	4	2	15	04	06	2
27	4	7	1	5	2	15	04	07	2
30	1	2	2	6	2	01	07	10	2
31	1	2	2	7	2	05	07	11	2
32	1	6	2	6	2	05	07	10	2
33	1	6	2	7	2	11	07	11	2
* 34	5	3	3	6		11	07	16	
* 35	5	3	3	7		15	07	17	
* 36	5	7	3	6		15	07	16	
* 37	5	7	3	7		01	07	17	

Fig. 9. rij en kolomnummers van machine E.

De partities gemerkt met * hebben meer dan twee toestanden in een blok.

inputvariabele X verkregen wordt.

Om een machine M te realiseren is men in de keuze van het aantal secundaire variabelen vrij.

Vaak zal men echter met het minimale aantal willen volstaan.

Bij deze behandeling zullen we ons daartoe beperken.

Voordat we het algorithme zullen behandelen voeren we nog eerst een aantal definities in, waarbij $\bar{\tau}$ de restrictie van τ is.

<u>Definitie 5.1.a.</u>

Van een machine M is $(\tau_1 \ldots \tau_m)$ een verzameling van tweebloksparitities op de verzameling van toestanden.

Een verzameling van partities $(M_{s-s}(\tau_1)\ldots M_{s-s}(\tau_k))$ heet $r_0 | (a_1 \ldots a_k)$-<u>toelaatbaar</u> of <u>streng toelaatbaar ten aanzien van de toestanden</u>, indien voor $k \leq r_0 \leq m$ en voor $i \in (1\ldots k)$ geldt:

a) $$\prod \bar{\tau}_j \leq M_{s-s}(\quad)$$

waarbij het aantal τ_j's dat nodig is voor de geldigheid van de ongelijkheid kleiner of gelijk is aan a_i:

b) $$\prod_{r_0} \tau_j = 0$$

waarbij r_0 het minimale aantal toestandsvariabelen is dat nodig is voor de realisatie van de machine M.

<u>Definitie 5.1.b.</u>

Van een machine M is $(\mu_1 \ldots \mu_n)$ een verzameling van tweebloksparitities op de inputs.

Een verzameling van partities $(M_{s-s}(\tau_1)\ldots M_{s-s}(\tau_k))$ heet $r_0 | (c_1 \ldots c_k)$-<u>toelaatbaar</u> of <u>strengtoelaatbaar</u> ten aanzien van de toestanden in de inputs indien voor $k \leq r_0 \leq m$ en voor $i \in (1\ldots k)$ geldt:

a) $$\prod \mu_j \leq M_{i-s}(\tau_i)$$

waarbij het aantal μ_j's dat <u>nodig</u> is voor de geldigheid van de ongelijkheid kleiner of gelijk is aan een getal b_i.

b) De verzameling $(M_{s-s}(\tau_1)\ldots M_{s-s}(\tau_k))$ $r_0 | (c_1-b_1, c_2-b_2, \ldots c_k-b_k)$-toelaatbaar ten aanzien van de toestanden is.

Indien we een toekenning doen van inputs aan een of meer input variabelen x_i met behulp van een partitie μ_i op de verzameling van inputs, dan is het aantal input-variabelen om de verschillende blokken van de input-partitie nog in de toewijzing te kunnen weergeven gelijk aan b_i;

$$b_i = \overline{\log}(\beta(\mu_i))$$

Hierbij is:

a) $\beta(\mu_i)$ = het aantal blokken van μ_i.

b) $\overline{\log}(x) = \log_2(x)$ naar boven afgerond op hele waarden; bijv. $\overline{\log}(4)=2$; $\overline{\log}(5)=3$.

Als we nu de tweebloksparitities, die moeten voldoen aan de eis uit 5.1.b: $\prod \mu_j \leq M_{i-s}(\tau_i)$, vrij mogen kiezen dan is het aantal partities μ_j dat we <u>nodig</u> hebben om te voldoen aan de ongelijkheid:

$$b_i = \overline{\log}(\ (M_{i-s}(\tau_i)))$$

Om nu een geschikte toewijzing te vinden is het dus zaak om een aantal τ_i te selekteren z.d.d. $(M_{s-s}(\tau_1)\ldots M_{s-s}(\tau_k))$ streng toelaatbaar is met $(a_1\ldots a_k)$ of, indien de inputs worden meegenomen, $(c_1\ldots c_k)$ zo gunstig mogelijk is.

We noemen $(a_1\ldots a_k)$ of $(c_1\ldots c_k)$ het <u>profiel</u>.

Het is nu de vraag welke verzameling $(a_1 \ldots a_k)$ of $(c_1 \ldots c_k)$ zo gunstig mogelijk is.
Indien een oplossing bestaat met alle $a_i=1$ dan is dit natuurlijk de meest gunstige (het combinatorische geval waarbij $a_i=0$ laten we buiten beschouwing).
Het criterium waarbij $\sum a_i$ zo klein mogelijk is, is over het algemeen niet zeer gunstig, omdat hierbij zich gevallen kunnen voordoen waarbij een van de a_i erg groot is ten opzichte van de andere;
een profiel volgens (2,2,2) zal de voorkeur verdienen boven (1,2,3). Omdat veel j-k-flipflops reeds met een combinatiemogelijkheid van inputs zijn uitgevoerd, die slechts een beperkte hoeveelheid combinatorische techniek kunnen verwerken, zal men over het algemeen bij een gelijkmatige verdeling met minder componenten de machine kunnen realiseren.
In het algorithme voor het bepalen van een oplossing die streng toelaatbaar is, bepaalt men eerst het profiel en daarbij zoekt men indien mogelijk $(M_{s-s}(\tau_1) \ldots M_{s-s}(\tau_k))$.
In fig. 10 staat vermeld in welke volgorde men het profiel op een gunstige manier kan doorlopen.
Indien een andere volgorde (i.v.m. de kostenfunctie) wordt gewenst, dan kan deze altijd ingevoerd worden.
Voorbeeld: Beschouw machine F (fig. 6b) en de bijbehorende rijnummers (fig. 9).
We gaan testen of toewijzing $17(\tau_1)$ en $33(\tau_2)$ een $2\,|(1,2)$-toelaatbare oplossing voor $(M_{s-s}(\tau_1), M_{s-s}(\tau_2))$ opleveren.

$\tau_1=(001111)$ dus $\overline{\tau_1}=(0011)$; $\tau_2=(011011)$ dus $\overline{\tau_2}=(0110)$; $\overline{\tau_1}.\overline{\tau_2}=0$.

$S33=(1,6,2,7)$ dus $M(\tau_2)=(0123)$ dus $M(\tau_2) \geqslant \overline{\tau_1}.\overline{\tau_2}$

S17=(5,5,3,3) dus $M(\tau_1)=(0011)=\overline{\tau}_1$.

Dit betekent dat $(M_{s-s}(\tau_1), M_{s-s}(\tau_2))$ streng toelaatbaar is ten aanzien van de toestanden met als profiel (1,2).

111	gunstig
112	
113	
222	
123	
223	
133	
233	
333	slecht

fig. 10a. Volgorde van te doorlopen profielen voor een machine met 3 toestandsvariabelen Y.

11	gunstig
12	
22	
13	
04	
23	
14	
33	
24	
34	
44	slecht

fig.10b. volgorde van te doorlopen profielen voor een machine met 2 toestandsvariabelen en 2 inputvariabelen.

Voorlopig houden we aan dat:

a) De som van de a_i's(c_i's) zo klein mogelijk moet zijn.

b) de "spreiding" van de a_i's(c_i's) zo klein mogelijk moet zijn.

We gaan nu proberen een zo gunstig mogelijk profiel voor machine E (fig. 6b) te vinden.

Hiertoe schrijven we alle tweebloksparties τ die voor een toekenning in aanmerking komen op.

Indien $\overline{\tau}$ meer dan 2 toestanden in een blok bevat dan kan geen streng toelaatbare oplossing meer gevonden worden. In fig. 9 zijn de toekenningen die meer dan 2

toestanden in een blok hebben, gemerkt met een "*".
Met behulp van de rij en kolomnummers kunnen we $M_{s-s}(\tau)$ en $M_{i-s}(\tau)$ vinden.
Afhankelijk van het aantal toestanden in het grootste blok van $M_{s-s}(\tau)$ en $M_{i-s}(\tau)$ kunnen we bepalen van hoeveel variabelen $v(\tau)$ <u>tenminste</u> afhankelijk is.
We rangschikken nu de toekenningen naar dit aantal.
Voor een afhankelijkheid van minimaal 3 variabelen vinden we voor machine E slechts toekenning (17).
Alle andere toekenningen zijn afhankelijk van 4 variabelen.
Voor toekenning (17) geldt:
$\tau_1=(001111)$; $\overline{\tau_1}=(0011)$; $M_{s-s}(\tau_1)=(0011)$; $M_{i-s}(\tau_1)=(123)$,
dus Y_1 is afhankelijk van zichzelf en van 2 inputs.
Meer toekenningen met een minimale afhankelijkheid van 3 variabelen waren er niet , dus het meest gunstige profiel wordt dan (3,4).
Een oplossing wordt dan gevonden door bij τ_1 nog een τ_2 te zoeken z.d.d. $\overline{\tau_1}.\overline{\tau_2}=0$.
(24) voldoet hieraan; $\overline{\tau_2}=(0101)$.
$(M_{s-s}(\tau_1), M_{s-s}(\tau_2))$ is nu streng toelaatbaar met als profiel (3,4).
Wat de verminderde afhankelijkheid betreft is de realisatie met toekenningen (17) en (24) optimaal.

5.4. Samenvatting van het algorithme.

1) Bereken alle kolom en rijnummers.

2) Bepaal welke tweebloksparttities τ niet voor een toekenning in aanmerking komen.
Indien een machine gerealiseerd kan worden met minimaal r_0 variabelen Y dan moet aan de volgende ongelijkheid worden voldaan:

$$\#(\overline{\tau}) \leq 2^{r_0-1}$$

waarbij $\#(\overline{\tau})$ betekent: het aantal toestanden in het grootste blok van $(\overline{\tau})$.

3) Uit de rijnummers kan het minimale aantal toestandsvariabelen waarvan de betreffende Y afhankelijk is worden bepaald. Dit aantal is $\overline{\log}(\beta(M_{s-s}(\tau)))$, waarbij $\beta(M_{s-s}(\tau))$ het aantal blokken van $M_{s-s}(\tau)$ voorstelt.

4) Uit de kolomnummers kan het minimale aantal inputvariabelen waarvan de betreffende Y afhankelijk is, worden bepaald.
Dit aantal is: $\overline{\log}(\beta(M_{i-s}(\tau)))$.

5) Het totale aantal variabelen waarvan een bepaalde Y afhankelijk is wordt dan minimaal:
$\overline{\log}\ (\beta(M_{i-s}(\tau)))+\overline{\log}(\beta(M_{s-s}(\tau)))=i.$

6) De nummers behorende bij de toekenningen; de rijnummers en de kolomnummers worden nu geselekteerd naar het totale minimale aantal variabelen waarvan de bybehorende

v afhankelijk is en in aparte rijen gezet(δ_i-rij).

7)Nu wordt het beginprofiel bepaald door voor de laagste i het aantal nummers in de δ_i-rij te tellen.
Is het aantal kleiner dan r_0 dan gaan we over naar de volgende δ_i-rij en j zo klein mogelijk en j>i enz.
Staan in de δ_i-rij 2 nummers in de δ_j-rij meer dan $(r_0-2)-1$ nummers dan wordt het beginprofiel voor $r_0=4$ (i,i,j,j).

8)We gaan nu na of er een aantal toekenningen zijn behorende bij het verkregen profiel, waarvan de τ_K's voldoen aan $\prod \overline{\tau}_K=0$.

9)Vinden we een combinatie van τ_K's die een doorsnede 0 opleveren dan gaan we na hoeveel partities τ_ℓ we <u>nodig</u> hebben om te voldoen aan: $\prod \tau_\ell \leq M_{s-s}(\tau_K)$.
Is dit aantal groter dan PROFIEL(k)-$\overline{\log}(\beta(M_{i-s}(\tau_K)))$, dan betekent dit dat de verzameling van $M_{s-s}(\tau_K)$'s niet streng toelaatbaar is tav. de input en de toestanden en we zoeken naar de volgende combinatie van τ_K's die voldoen aan $(\overline{\tau}_K)=0$.

10)Vinden we een combinatie van τ_K's die voldoet aan de eisen onder 8) en is het aantal partities τ_ℓ dat we nodig hebben om te voldoen aan $\prod \overline{\tau}_\ell \leq M_{s-s}(\tau_K)$ gelijk aan PROFIEL(k)-$\overline{\log}(\beta(M_{i-s}(\tau_K)))$ voor alle τ_K, dan kunnen we met behulp van de τ_K's een toekenning doen die een maximaal verminderde afhankelijkheid geeft.

5.5. Conclusie.

In de appendix vindt u een aantal geprogrammeerde algorithmen(PL/1):

1) Een algorithme voor het berekenen van de $M_{s-s}(\tau)$ en de $M_{i-s}(\tau)$ partities waarbij τ een tweebloksparttitie is. Dit algorithme moet gezien worden als een inleiding voor het algorithme dat de toewijzing berekent voor de realisatie van een machine.

2) Een algorithme waarbij de toewijzingen afhankelijk van het minimale aantal afhankelijke variabelen, zijn geselecteerd.
I.v.m. de benodigde geheugenruimte is dit algorithme verder niet toegepast.

3) Een algorithme dat achtereenvolgens verschillende profielen doorloopt van gunstig tot ongunstig voor $r_0=2,3$ en 4.
Als het maximale aantal afhankelijkheden is hierbij 4 aangenomen.

4) Het hoofdalgorithme dat in staat is de meest gunstige toekenning(tav. de verminderde afhankelijkheid) te berekenen.
Het programma kan een machine met maximaal 15 toestanden en 15 inputs verwerken.
Het aantal toekenningen dat in aanmerking komt voor een eventuele realisatie mag echter niet meer dan 880 bedragen.

De rekentijd is afhankelijk van de te realiseren machine; bestaat er een oplossing met een gunstig profiel dan wordt de oplossing snel gevonden, oplossingen met minder gunstige profielen vergen meer rekentijd. Voor alle berekende voorbeelden was de rekentijd echter gering in verhouding tot de vertaaltijd.

6. Het realiseren van een machine met behulp van j-k-flipflops.

6.1. Inleiding.

Bij het criterium van de verminderde afhankelijkheid kan men nog een verdere splitsing aanbrengen indien gebruik wordt gemaakt van j-k-flipflops.

Machine G (fig. 11) is een goed voorbeeld om dit te illustreren.

	x
1	2
2	3
3	1

fig. 11. Machine G.

De rijnummers behorende bij de toekenningen zijn:

(1) rijnummer: (0,1,0)

(2) (1,0,0)

(3) (1,1,0)

We kunnen de machine realiseren met 2 variabelen: (v_1,v_2), (v_1,v_3) of (v_2,v_3). Hierbij hoort v_i bij toekenning (i).

Voor de eerste realisatie vinden we de volgende afhankelijkheden: $Y_1=f(v_2)$, $Y_2=f(v_1,v_2)$.

Voor de 2^e realisatie vinden we: $Y_3=f(v_3)$, $Y_1=f(v_1,v_3)$.

En voor de laatste realisatie: $Y_2=f(v_3)$, $Y_3=f(v_2,v_3)$.

Voor geen van deze 3 toekenningen bestaat een voorkeur.

We gebruiken nu flipflops overeenkomstig de tabel in fig. 12.

De inverse functie tabel is eenvoudig met behulp van fig. 12 af te leiden.

We doen nu alle mogelijke toekenningen voor machine G overeenkomstig hoofdstuk 5 en laten deze in de tabel staan.

j	k	v_{t+1}
0	0	v_t
0	1	0
1	0	1
1	1	$\overline{v}_t$

fig. 12. functietabel van een j-k-flipflop

v_t	v_{t+1}	j	k
0	0	0	-
0	1	1	-
1	0	-	1
1	1	-	0

fig. 13. inverse functietabel van een flipflop. (-=don't care)

Achter de tabel zetten we nu nog 2 tabellen ; een voor de j-ingang en een voor de k-ingang van de flipflop. (zie fig. 14).

v_t	v_{t+1}	$v_1(t)$	$(t+1)$	j_1	k_1	$v_2(t)$	$(t+1)$	j_2	k_2	$v_3(t)$	$(t+1)$	j_3	k_3
1	2	0	0	0	-	0	1	1	-	0	1	1	-
2	3	0	1	1	-	1	0	-	1	1	1	-	0
3	1	1	0	-	1	0	0	0	-	1	0	-	1

Fig. 14. Verschillende toekenningen voor machine G.

De eerste realisatie levert op:

voor v_1 $\quad j_1=v_2 \quad k_1=1$

voor v_2 $\quad j_2=\overline{v}_1 \quad k_2=1$

En voor de 2[e] realisatie:

$v_1 \quad j_1=v_3 \quad k_1=1$

$v_3 \quad j_3=1 \quad k_3=v_1$

En voor de laatste realisatie:

$v_2 \quad j_2=\overline{v}_3 \quad k_1=1$

$v_3 \quad j_3=1 \quad k_3=\overline{v}_2$

We zien duidelijk dat de afhankelijkheden sterker verminderd zijn dan de berekende afhankelijkheden in het algemene geval.

We zullen nu overgaan tot het beschrijven van een algorithme voor een verminderde afhankelijkheid tav. de j en k ingang bij toepassing van j-k-flipflops.

6.2. Het genereren van rij- en kolomnummers voor de j en de k ingang afzonderlijk.

Om rij en kolomnummers te generen kan een zeer eenvoudige procedure worden toegepast.

We staan hierbij d.c. condities toe.

We mogen nu echter niet de rij en kolomnummers genereren door optellen van reeds eerder gevonden rijnummers of kolomnummers; de volgorde tabel moet voor elke nieuw toekenning ingevuld worden om de betreffende nummers te bepalen.

Met behulp van de inverse functietabel van een j-k-flipflop(fig. 13) wordt dan voor de j- ingang en voor de k-ingang een tabel verkregen.(Zie machine G fig. 14)Voor machine G zijn nu de rij en kolomnummers:

(1) Sj1=(0,1,-); Sk1=(-,-,1)

(2) Sj2=(1,-,0); Sk2=(-,1,-)

(3) Sj3=(1,-,-); Sk3=(-,0,1).

Voor de kolomnummers ligt de zaak wat moeilijker; de nummers worden binair opgeschreven om nog aan te kunnen geven welke plaatsen d.c. zijn.

(1) Ij1=(01-) Ik1=(--1)

(2) Ij2=(1-0) Ik2=(-1-)

(3) Ij3=(1--) Ik3=(-01)

Terwille van de duidelijkheid geven we nog een voorbeeld van een toewijzing voor een machine met meerdere inputs.

Beschouw machine E (fig, 6b) met de toewijzing (14).

Dan ziet de tabel er uit als fig. 15.

	a	b	c	a	b	c	a	b	c
0	1	0	1	1	0	1	-	-	-
0	0	0	1	0	0	1	-	-	-
1	0	1	1	-	-	-	1	0	0
1	0	1	0	-	-	-	1	0	1
				j-ingang			k-ingang		

fig. 15. Toewijzing(14)voor machine E.

Sj14=(5,1,-,-);   Sk14=(-,-,4,5)

We leggen nu beperkingen op voor de toewijzing van de d.c. in de rijnummers. Voor de d.c.'s mogen slechts de reeds vastgelegde rijnummers worden gebruikt.

Dit wordt aangegeven door een *. Dus Sj14=(5,1,*,*) en Sk14=(*,*,4,5).

Voor Sk14 kan voor een * dus uitstluitend een 4 of een 5 worden gekozen.

Nà deze beperking zijn we in staat de kolomnummers eenduidig te noteren.

Hiertoe intróduceren we eerst een hulpstelling:

Stelling 6.1.

Indien in de rijnummers beperkte don't cares voorkomen kunnen de kolomnummers eenduidig genoteerd worden.

De rijnummers die geen d.c.zijn geven voldoende informatie over het al dan niet gelijk zijn van de kolomnummers.

Bewijs:

Beschouw de tabel met o.a. de beperkte d.c. en beschouw de rijnummers die geen d.c. zijn als vectoren:

$\underline{b}_i=(b_{i1},\ldots b_{in})$, met $b_{ij}=$ "0" of "1".

De tabel ziet er dan als volgt uit:

$$
\begin{array}{l}
b_{11}\ldots b_{1j}\ldots b_{1n} \\
\ldots\ldots\ldots\ldots\ldots \\
* \quad \ldots * \quad \ldots * \\
b_{i1}\ldots b_{ij}\ldots b_{in} \\
* \qquad * \qquad * \\
\ldots\ldots\ldots\ldots\ldots \\
b_{k1}\ldots b_{kj}\ldots b_{kn}
\end{array}
$$

De kolomnummers worden bepaald door de kolomvectoren

$\underline{b}_j=(b_{1j},\ldots b_{ij},\ldots b_{kj})$.

Een rij met beperkte d.c. krijgt nu een waarde toegewezen bijv. $\underline{b}_i$.

Stel nu dat de kolomvectoren $\underline{b}_j$ en $\underline{b}_m$ gelijk zijn de nieuwe kolomvectoren : $(b_{1j}\ldots \overset{*}{1},\ldots b_{ij},\ldots b_{kj})=$ $(b_{1m},\ldots \overset{*}{2},\ldots b_{im},\ldots b_{km})$ ook gelijk want $\overset{*}{1}=b_{ij}$ en $\overset{*}{2}=b_{im}$ en $b_{ij}=b_{im}$.

De kolomnummers worden dus bepaald door de rijnummers die niet d.c. zijn. Q.E.D.

Voor fig. 15 (toewijzing (14))zijn de kolomnummers dan: Ij14=(2,0,3) en Ik14=(3,0.1).

6.3. Het bepalen van $M^j(\tau)$ en $M^k(\tau)$.

Om te bepalen van welke variabelen j en k van een flipflop afhankelijk zijn, definieren we een aantal nieuwe partities: $M^j_{s-s}(\tau)$; $M^k_{s-s}(\tau)$; $M^j_{i-s}(\tau)$ en $M^k_{i-s}(\tau)$.

Definitie6.1.

Indien geldt dat: Sj1(k)=Sj1(h);

Sk1(k)=Sk1(h);

Ijv(u)=Ijv(w);

Ikv(u)=Ikv(w);

dan komen de toestanden k en h en de inputs u en v in hetzelfde blok voor van de overeenkomstige $M(\tau)$ partitie.

Indien in de rijnummers een d.c. (*) voorkomt dan wordt deze ook als d.c. meegenomen naar de betreffende $M(\tau)$ partitie.(Schrijfwijze: de toestanden liggen vast en de blokken worden genoteerd)

Alleen het blok waarin de overeenkomstige toestand zich bevindt is dan een d.c.

De $M(\tau)$ partities behorende bij toewijzing (14) van machine F (fig. 15) zijn dan:

Sj14=(5,1,*,*) dus $M^j_{s-s}(\tau)$=(1,2,*,*)

Sk14=(*,*,4,5) $M^k_{s-s}(\tau)$=(*,*,1,2)

Ij14=(2,0,3) $M^j_{i-s}(\tau)$=(1,2,3)

Ik14=(3,0,1) $M^k_{i-s}(\tau)$=(1,2,3)

Het gebruik van beperkte d.c. legt ons ook beperkingen op tav. het blok waarin een bepaalde toestand zich bevindt. Deze toestand mag zich slechts in een van de reeds aanwezige blokken van $M_{s-s}(\tau)$ bevinden.

In de $M_{i-s}(\tau)$ partities komen geen d.c.'s voor of alle toestanden komen in een d.c. blok voor.
Ook bij $M_{s-s}(\tau)$ kunnen alle toestanden in een d.c. blok voorkomen.
In het kader van dit algorithme is het gunstig om voor deze d.c. partities de eenheidspartitie I te kiezen.

Om nu na te gaan in welke mate verminderde afhankelijkheid optreedt voor de j en de k ingang van de flipflop definieren we nog de volgende ongelijkheid:
Definitie 6.2.
Beschouw een partitie π en een partitie $M(\tau)$ op een verzameling van toestanden S,waarbij in $M(\tau)$ een aantal toestanden in een beperkt d.c. blok kunnen voorkomen. Dan is $\pi \leq M(\tau)$ indien voor alle toestanden s en t die voldoen aan $s \equiv t(\pi)$ geldt dat:
of $s \equiv t(M(\tau))$,
of s en/of t komen in een beperkt d.c.blok voor.
De tweede voorwaarde zal volkomen duidelijk zijn indien we bedenken dat voor $\pi = \prod_{i \in P_j} \tau_i$ geldt dat $\pi \leq M(\tau)$ het aantal elementen van P_j kleiner kan zijn indien we de d.c. blokken op een gunstige manier kiezen.

De stelling betreffende de verminderde afhankelijkheid is volledig toepasbaar op de j en de k ingang van de flipflops.
De definities over de strenge toelaatbaarheid worden voor de j-k-flipflop analoog als die onder 5.1 echter met dien verstande dat a_i, b_i en c_i nu vectoren worden van de dimensie 2 .

Hiervan slaat de eerste term op de j ingang en de laatste term op de k ingang. Het profiel wordt nu ook gevormd door vectoren en kan er als volgt uitzien:

$$((c_{j1},c_{k1}),(c_{j2},c_{k2}),(c_{j3},c_{k3}))$$

Dit betekent dat er naar een realisatie gezicht moet worden waarbij de eerst flipflop c_{1j} variabelen (toestand en input) aan de j ingang krijgt toegevoerd en c_{1k} variabelen aan de k ingang enz.

6.4 Het profiel.

Evenals in hoofdstuk 5 zal een profiel gekozen moeten worden dat zo gunstig mogelijk is.
Bij dit profiel wordt nagegaan of er een streng toelaatbare oplossing bestaat.
Hieruit volgt dat de profielen het best doorlopen kunnen worden in de volgorde van "gunstig " tot "ongunstig". Hiervoor worden de volgende criteria aangelegd:

1) Het totaal van het aantal afhankelijkheden moet zo klein mogelijk zijn; dwz. $(c_{1j}+c_{1k}+c_{2j}....)$ is zo klein mogelijk.
2) Bestaat er een keuzemogelijkheid tussen twee profielen waarvan het totaal gelijk is dan wordt die mogelijkheid gekozen waarbij $c_{1j}+c_{1k},c_{2j}+c_{2k},.....$ een zo gelijkmatig mogelijke verdeling heeft.
3) Bestaat er nog een keuzemogelijkheid waarbij $(c_{ij}+c_{ik})$'s gelijk zijn, dan wordt die c_{ij} en c_{ik} gekozen waarvan de verdeling zo gelijmatig mogelijk is.

6.5. Voorbeeld van het verloop van het algorithme.

Een samenvatting van het verloop van het algorithme zal ongeveer hetzelfde zijn als de samenvatting beschreven in hoofdstuk 5.

We zullen echter nog een zeer korte samenvatting geven; alleen daar waar belangrijke verschillen optreden zullen we wat uitgebreider zijn.

Samenvatting:

1) Genereer alle j en k kolom- en rijnummers voor de in aanmerking komde toewijzingen.

2) Bepaal hieruit all $M(\tau)$'s.
In feite is dit al gedaan; volgens de nieuwe schrijfwijze hernummeren we slechts de rijnummers.

3) Bepaal voor elke j en k input hoeveel variabelen minstens moeten worden toegevoerd.(secundaire variabelen en inputs)
Tel voor elke flipflop(toekenning) het aantal variabelen op (totaal voor j en k ingang).
Het totaal is i.

4) De som van de optelling uit 3 geeft aan in welke δ_i-rij de toekenningen komen te staan.
Bovenaan in een δ_i-rij komen de "gunstige" afhankelijkheden te staan; onderaan de "ongunstige".
Dit is voor een bepaalde δ_i-rij alleen afhankelijk van het verschil tussen de minimaal toe te voeren variabelen aan de j en de k ingang.
Hoe groter het verschil, des te "ongunstiger" zijn deze toekenningen.

5) Nu wordt het beginprofiel bepaald; we gaan na of er

toekenningen zijn die voor de bijbehorende partities een doorsnede nul opleveren.

Is dit niet het geval dan nemen we het volgende profiel, enz.,totdat de doorsnede bij een bepaald profiel wel nul oplevert.

6)We controleren of de afhankelijkheden overeenstemmen met het tot nu toe gekozen meest gunstige profiel.

Is dit niet zo dan proberen we alle andere combinatiemogelijkheden in overeenstemming met dit profiel,enz.

toekenning	rijnummers J 1	J 2	J 3	J 4	K 1	K 2	K 3	K 4	J a	J b	J c	K a	K b	K c	aantal var S j	S k	I j	I k	tot. aantal j	tot. aantal k	tot. aantal j+k	volgorde
14	5	1	*	*	*	*	4	5	2	0	3	3	0	1	1	1	2	2	3	3	6	11
15	5	5	*	*	*	*	4	4	2	0	3	3	0	0	1	0	2	1	3	1	4	4
16	5	5	*	*	*	*	4	5	3	0	3	3	0	1	0	1	1	2	1	3	4	5
17	5	*	*	*	*	*	4	4	3	0	3	3	0	0	0	0	1	1	1	1	2	1
24	4	*	1	*	*	4	*	3	2	0	1	2	1	1	1	1	2	1	3	2	5	6
25	4	*	1	*	*	4	*	2	2	0	1	2	1	0	1	1	2	2	3	3	6	12
26	4	*	1	*	*	0	*	3	2	0	1	0	1	1	1	1	2	1	3	2	5	7
27	4	*	1	*	*	0	*	2	2	0	1	0	1	0	1	1	2	1	3	2	5	8
30	1	*	*	6	*	5	5	*	1	1	2	3	0	3	1	0	1	1	2	1	3	2
31	1	*	*	7	*	5	5	*	1	1	3	3	0	3	1	0	1	1	2	1	3	3
32	1	*	*	6	*	1	5	*	1	1	2	1	0	3	1	1	1	2	2	3	5	9
33	1	*	*	7	*	1	5	*	1	1	3	1	0	3	1	1	1	2	2	3	5	10

Fig. 16. Tabel behorende bij machine E.

In de tabel van fig. 16 staan alle gegevens die we nodig hebben voor het doen van een toekenning.

In de rij j+k staat het totale aantal variabelen dat minstens aan de input van een j-k-flipflop moet worden toegevoegd.

Hiermee kunnen we de δ_i-rijen bepalen (fig. 17).

δ_2	δ_3	δ_4	δ_5	δ_6
17	30	15	24	14
	31	16	26	25
			27	
			32	
			33	

Fig. 17. δ_i-rijen behorende bij machine E.

Het beginprofiel wordt: (2,3).

De eerste twee toekenningen die voor de bijbehorende partities een doorsnede nul opleveren zijn (17) en (30).

$\tau_1=(001111)$ dus $\overline{\tau}_1=(0011)$

$\tau_2=(011000)$ dus $\overline{\tau}_2=(0110)$

$M^j_{s-s}(\tau_1)=(11{*}{*})=I$; $M^k_{s-s}(\tau_1)=({*}{*}11)=I$.

Hieruit volgt dat Y_1 onafhankelijk is van y_1 en y_2.

$j_1=f(\underline{x})$; $k_1=f(\underline{x})$.

$M^j_{s-s}(\tau_2)=(1{*}{*}2)$ en $M^k_{s-s}(\tau_2)=({*}11{*})=I$.

$M^j_{s-s}(\tau_2)$ dus $j_2=f(y_1,\underline{x})$ en $k_2=f(\underline{x})$.

De oplossing is nu streng toelaatbaar en het profiel (2,3) kan worden gerealiseerd.

Een toewijzing voor $\underline{x}$ volgt uit de kolomnummers.

Voor (17) levert dit een toewijzing volgens figuur 18a op.

	x_1	x_2			x_3	x_4			x_1	x_2	x_3
a	1	1		a	0	1		a	1	1	0
b	0	0		b	0	0		b	0	0	0
c	1	0		c	1	1		c	1	0	1
	a)				b)				c)		

Fig. 18. Toewijzingen van de inputs voor machine E.

Voor (24) levert dit een toewijzing volgens fig. 18b op.

We zien dat x_4 gelijk is aan x_1; dus de toewyzing voor x_4 is overbodig.

Resumerend:

Voor de eerste flipflop volgt nu: $j=f(x_1)$; $k=f(x_2)$.

Voor de tweede flipflop volgt : $j=f(y_1,x_3)$; $k=f(x_1)$.

Met behulp van een Karnaugh-diagram zijn de logische vergelijkingen exact te bepalen.

$j_1=x_1$; $k_1=x_2$; $j_2=y_1.\overline{x}_3+\overline{y}_1.x_3$; $k_2=x_1$.

De realisatie van de schakeling ziet er dan uit volgens fig. 19.

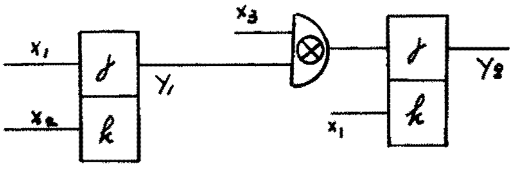

Fig. 19. Realisatie van machine E.

Ook de toewijzingen(17) en (31) zouden een geschikte oplossing geven.

De toewijzingen voor <u>x</u> blijven dezelfde.

Er volgt dan voor de j en de k ingangen:

$j_1=x_1$; $k_1=x_2$; $j_2=y_1+x_3$; $k_2=x_3$.

Hieruit volgt dat de exclusive-or in fig. 19 ook vervangen mag worden door een or-schakeling.

6.6. Conclusie.

In hoofdstuk 5 was de toewijzing (17) en(24) volgens het daar gestelde criterium een gunstige toewijzing. Zouden we de schakeling realiseren met flipflops overeenkomstig deze toewijzing dan betekent dit dat aan de j_2 ingang minstens 3 variabelen, en aan de k_2 ingang minstens 2 variabelen toegevoerd moeten worden. Algemeen kunnen we dan ook stellen dat indien de schakeling gerealiseerd gaat worden met j-k-flipflops, het algorithme uit hoofdstuk 6 een betere oplossing geeft.

Bij het opstellen van het algorithme is rekening gehouden met het feit dat het geheel direct te programmeren moet zijn. Het geprogrammeerde algorithme, beschreven in hoofdstuk 5, kon daarvoor goed als uitgangspunt dienen.

7. Toewijzingen bij een van te voren vastliggende input.

In vele gevallen zijn we niet meer vrij in de keuze van een input-toewijzing omdat de input reeds in een bepaalde vorm aangeboden wordt.
De algorithmen beschreven in de hoofdstukken 5 en 6 zullen dan een kleine wijziging moeten ondergaan om toch nog tot een optimaal verminderde afhankelijkheid te komen.

Stel dat de input is toegewezen aan de hand van een aantal partities μ_j en de toestanden aan de hand van een aantal partities τ_i.
Een verzameling van indices, te gebruiken om onderscheid te maken tussen de verschillende μ_i's, noemen we O_i.
We zijn nu in staat de volgende stelling te formuleren.

Stelling 7.1.

Van een machine M zijn $(\mu_1,\ldots,\mu_j,\ldots\mu_K)$ partities op de verzameling van inputs en $(\tau_1,\ldots,\tau_i,\ldots\tau_K)$ partities op de verzameling van toestanden.
Indien geldt dat:

a) $$\prod_{j\in O_i} \mu_j \leq M_{i-s}(\tau_i)$$

b) Het aantal elementen van O_i is zo klein mogelijk

dan volgt voor de afhankelijkheid van Y_i t.a.v. de inputs x_i:

$Y_i = f(\underline{y},\ldots x_m, x_n\ldots)$ met m,n O_i.

Met deze stelling zijn we dus in staat de afhankelijkheden van Y_i (hoofdstuk 5), j_i of k_i (hoofdstuk 6) na te gaan t.a.v. de input.

Voorbeeld: machine E (fig 6b).

We proberen een zo goed mogelijke toewijzing te doen met behulp van j-k-flipflops.

In de tabel van fig.16 hoeven we daartoe alleen de laatste 6 kolommen te veranderen.

Stel nu dat de input is toegewezen volgens fig. 20.

	x_1	x_2
a	0	0
b	0	1
c	1	1

fig. 20. Input toewijzing voor machine E.

De partities behorende bij x_1 en x_2 zijn dan:

$\mu_1 = (\overline{a,b};\overline{c})$ of volgens de nieuwe schrijfwijze (001).

$\mu_2 = (\overline{a};\overline{b,c})$ (011).

Met behulp van de kolomnummers ($M_{i-s}(\tau)$) is dan de afhankelijkheid van j en k t.a.v. de input te bepalen.

De resultaten staan in de tabel van fig. 21.

Deze resultaten worden verder verwerkt overeenkomstig het algorithme uit hoofdstuk 6; voor machine E zal in dit geval de toestandstoewijzing hetzelfde blijven, nl. (17) en (30); voor een andere machine zou het resultaat echter geheel verschillend kunnen zijn.

toe ken ning	aantal var. S j	aantal var. S k	aantal var. I j	aantal var. I k	tot. aan tal j	tot. aan tal k	j+k	volg orde
14	1	1	2	2	3	3	6	10
15	1	0	2	1	3	1	4	4
16	0	1	2	2	2	3	5	5
17	0	0	2	1	2	1	3	1
24	1	1	2	1	3	2	5	6
25	1	1	2	2	3	3	6	11
26	1	1	2	1	3	2	5	7
27	1	1	2	2	3	3	6	12
30	1	0	1	2	2	2	4	2
31	1	0	1	2	2	2	4	3
32	1	1	1	2	2	3	5	8
33	1	1	1	2	2	3	5	9

δ-rijen:

δ_3	δ_4	δ_5	δ_6
17	30	16	14
	31	24	25
	15	26	27
		32	
		33	

Fig. 21. Tabel behorende bij machine F met vaste input.

8. Literatuurlijst.

[1] J. Hartmanis, R.E. Stearns, "Algebraic Strcture Theory of Sequential Machines", Prentice-Hall, Inc.

[2] R.E. Stearns, J. Hartmanis, "On the State Assigment Problem for Sequential machines II", IRE transactions on electronic computers, Vol. EC-10, no. 4 (december 1961), 593-603.

[3] Peter Weiner, Edward J. Smith, " Optimization of Reduced Dependencies for Synchronous Sequential Machines", IEEE transactions on electronic computers, Vol. EC-16, no. 6 (december 1967), 835-847.

```
************************************
*                                  *
* APPENDIX                         *
*                                  *
* BEHORENDE BIJ HET                *
* AFSTUDEERVERSLAG VAN             *
* P• G• JANSEN                     *
*                                  *
* DSLPROG     16/11/70    12•19•08 *
************************************
```

Inhoud

1• Berekening van Mm-paren•

Het hierna volgende programma berekent de Mm-paren van een willekeurige machine•
Het maximaal te verwerken aantal toestanden is 16, inclusief het aantal gespecificeerde don't cares•
Er kunnen niet meer dan 200 Mm-paren berekend worden•

Op de invoerkaarten van de te berekenen machine moeten de gegevens uit de volgorde tabel als volgt worden geponst:

1) Op de eerste twee plaatsen staat het aantal rijen (toestanden zonder d•c) van de volgordetabel; bijvoorbeeld 07•
2) Op de derde en de vierde plaats staat het aantal kolommen (inputs) van de volgordetabel; bijvoorbeeld 03•
3) op de vijfde en de zesde plaats staat het aantal toestanden zonder de gespecificeerde don't cares (zelfde aantal als onder 1));bijvoorbeeld 07•
4) Op de zevende en de achtste plaats staat het aantal toestanden inclusief de gespecificeerde don't cares•
5) Op de nu volgende plaatsen staat de inhoud van de volgordetabel van links naar rechts en van boven naar beneden• Elke toestand wordt door twee decimale getallen weergegeven; dus 6:=06•

```
// JOB PL/I,ECB,JANSENPG                    30/10/70 10-27-21 026   14•17•59
// OPTION LINK,LIST,SYM
// EXEC PL/I
```

4

O P T I O N S L I S T

OPTIONS TAKEN ARE 60C,LIST,SYM,ERRS,OPT•

```
PARTP: PROCEDURE OPTICNS(MAIN);

   1              PARTP: PROCEDURE OPTIONS(MAIN);
   2              DECLARE (PARTL(200,16),PARTR(200,16),PARTO(16),PARTN(16),PARTC(16),
                  PARTLN(16),PARTRN(16),TABLE(16,16),PAARL(2,16),BLOK(16),
                  BLCKNC,PARTNO) FIXED DECIMAL(2),
                  (GROPRTNO,GRNPRTNO,C,D,F,I,J,K,L,M,MR,ML,N,O) FIXED BINARY(5),
                  (GEHAD(16)) FIXED BINARY(1),
                            (V,W) FIXED DECIMAL(1),
                  (A,B,U,X,Y,Z,ZZ) FIXED BINARY(8),
                  PARTLPT(16) FIXED DECIMAL(2) BASED(PL),
                  PARTRPT(16) FIXED DECIMAL(2) BASED(PR),
                  (PL,PR) POINTER;

   3              CHEK: PROCEDURE;

                  /* VERGELIJK 2 PARTITIES, RESULTAAT: GROTER KLEINER, GELIJK OF ONGELIJK
                   */

   4                 V=0;
   5                 GRNPRTNO=PARTN(1); GROPRTNO=PARTO(1);
   7                 DO I=1 TO M;
   8                 IF GRNPRTNO<PARTN(I) THEN GRNPRTNO=PARTN(I);
   9                 IF GROPRTNO<PARTO(I) THEN GROPRTNO=PARTO(I);
  10                 END;
  11                 IF GRNPRTNO>GROPRTNO THEN DO;
  12                      DO I=1 TO GRNPRTNO;
  13                      DO J=1 TO M;
  14                      IF PARTN(J)=I THEN DO;
  15                           PARTNO=PARTO(J); GOTO DDD;
  17                                END;
  18                      END;
  19              DDD:
                     DO K=J+1 TO M;
  20                      IF PARTN(K)=I THEN DO;
  21                      IF PARTO(K)¬= PARTNO THEN DO; V=4; GOTO FFF; END;
  25                                END;
  26                 END;    END;
  28                 V=3;
  29              FFF: END;
  30                 ELSE DO;
  31                 DO I=1 TO GROPRTNO;
  32                 DO J=1 TO M;
  33                 IF PARTO(J)=I THEN DO;
  34                      PARTNO=PARTN(J);
  35                      GOTO EEE;
  36                                END;
  37                  END;
  38              EEE:
                      DO K=J+1 TO M;
  39                 IF PARTO(K)=I THEN DO;
  40                      IF PARTN(K)¬=PARTNO THEN DO; V=4; GOTO GGG; END;
  44                           END;
  45                 END;
  46                 END;
  47                      IF GRNPRTNO=GROPRTNC THEN V=2;
  48                 ELSE V=1;
```

```
  49             GGG: END;
  50             END CHEK;

  51             OPTL: PROCEDURE;

                 /* SOMMEER 2 PARTITIES */

  52             DECLARE (NOGDOENO(16),NOGDOENN(16),GEHADO(16),GEHADN(16))
                 FIXED DECIMAL(2),
                 (NDN,NDO,GO,GN) FIXED BINARY(5);
  53             IF GROPPTNO<3|GRNPRTNO<3 THEN DO;
  54             PARTC=1;
  55             GOTO EEM;
  56             END;
  57             ELSE DO;
  58                     I=1; PARTNO=1; NOGDOENO=00; NOGDOENN=00;NDN=0; PARTC=00;
  64                     GEHAD=0;
  65             III:
  66                     NDO=1;GEHADO=00;GO=0;GEHADN=00;GN=0;
  70                     NOGDOENO(1)=I;
  71             HHH:
                              DO K=1 TO NDO;
  72                               DO J=1 TO M;
  73                               IF PARTO(J)=NOGDOENO(K) THEN DO;
  74                                    GEHAD(J)=1;
  75                                         DO L=1 TO NDN;
  76                                         IF NOGDOENN(L)=PARTN(J) THEN GOTO KKK;
  77                                         END;
  78                                         DO O=1 TO GN;
  79                                         IF GEHADN(O)=PARTN(J) THEN GOTO KKK;
  80                                         END;
  81                                         NDN=NDN+1;
  82                                         NOGDOENN(NDN)=PARTN(J);
  83                                              END;
  84             KKK:              END;
  85                                    GO=GO+1;
  86                                    GEHADO(GO)=NOGDOENO(K);
  87                          END;
  88                     NDO=0;NOGDOENO=00;
  90                     IF NDN=0 THEN GOTO INSR;
  91                          DO K=1 TO NDN;
  92                               DO J=1 TO M;
  93                               IF PARTN(J)=NOGDOENN(K) THEN DO;
  94                                    GEHAD(J)=1;
  95                                    DO L=1 TO NDO;
  96                                    IF NOGDOENO(L)=PARTO(J) THEN GOTO LLL;
  97                                    END;
  98                                    DO O=1 TO GO;
  99                                    IF GEHADO(O)=PARTO(J) THEN GOTO LLL;
 100                                    END;
 101                                    NDO=NDO+1;
 102                                    NOGDOENO(NDO)=PARTO(J);
 103                                              END;
 104             LLL:              END;
 105                                    GN=GN+1;
```

```
 106                                    GEHADN(GN)=NOGDOENN(K);
 107                          END;
 108                      NDN=0; NOGDOENN=00;
 110                     IF NDO¬=0 THEN GOTO HHH;
 111         INSR:        J=GROPRTNO+1;
 112                     DO K=1 TO M;
 113                     IF GEHAD(K)=1 THEN DO;IF PARTC(K)=0 THEN
                                   PARTC(K)=PARTNO;
 115                               END;
 116                          ELSE DO;
 117                               IF PARTO(K)<J THEN DO;
 118                                    IF PARTO(K)>I THEN DO;
 119                                    J=PARTO(K);
 120                     END;      END;       END; END;
 124                     IF J>GROPRTNO THEN GOTO MMM;
 125                     ELSE I=J; PARTNO=PARTNO+1;
 127                     GOTO III;
 128         END;
 129         MMM:
             END OPTL;

 130         PRINT: PROCEDURE;

             /* PRINTEN VAN PARTITIEPAREN */

 131         PUT SKIP;
 132         PUT SKIP EDIT ('VOLGENDE GENERATIE MM-PAREN') (A);
 133         PUT SKIP;
 134         DO Y=A TO B;
 135         PL=ADDR(PARTL(Y,1)); PR=ADDR(PARTR(Y,1));
 137         PUT SKIP;
 138         M=ML; W=0; DO O=1 TO ML; PARTO(O)=PARTLPT(O); END;
 143         WEER:
                 PUT EDIT ('( 1')(A(3));
 144             GEHAD=0; GEHAD(1)=1; I=1;
 147         NOG:
                 DO O=I+1 TO M;
 148             IF PARTO(O)=PARTO(I) THEN DO;
 149                  GEHAD(O)=1;
 150                  PUT EDIT ('-',O)(A(1),F(2));
 151                            END;
 152             END;
 153             DO O=I+1 TO M;
 154             IF GEHAD(O)=0 THEN DO;
 155                  PUT EDIT (';',O)(A(1),F(2));
 156                  I=O;
 157                  GOTO NOG;
 158                            END;
 159             END;
 160             PUT EDIT (') ')(A(2));
 161             IF W=0 THEN DO;
 162                  W=1; M=MR;
 164                  DO O=1 TO MR; PARTO(O)=PARTRPT(O); END;
 167                  GOTO WEER;
 168                            END;
```

```
PARTP: PROCEDURE OPTICNS(MAIN);

 169                    END;
 170                         END PRINT;
 171                GET EDIT (M,N,ML,MR)(F(2));
 172                DC I=1 TC M; DO J=1 TO N; GET EDIT (TABLE(I,J))(F(2));
 175                PUT EDIT (TABLE(I,J))(X(2),F(2)); END; PUT SKIP; END;

                    /* EEREKINING VAN DE VOLGENDE BASISGENERATOR */

 179                Z=0;
 180                DO A=1 TO M-1;
 181                DC B=A+1 TO M;
 182                DO C=1 TO N;
 183                         PAARL(1,C) = TABLE(A,C);
 184                         PAARL(2,C) = TABLE(B,C);
 185                END;
 186                GEHAD=0; BLOKNO=1; PARTRN=00; PARTLN=00; D=1;
 191                BEB:
                    BLOK=00;
 192                BLCK(1)=PAARL(1,D);
 193                GEHAD(D)=1;
 194                F=1;
 195                 IF PAARL(2,D)¬=BLOK(1) THEN DO;
 196                  F=F+1;
 197                  BLOK(2)=PAARL(2,D);
 198                                          END;
 199                DC C=D+1 TO N;
 200                 K=C;
 201                AAA:
                       IF GEHAD(K)=1 THEN DO;
 202                     IF K<C THEN DO;
 203                       K=K+1; GOTO AAA;
 205                                 END;
 206                            ELSE GOTO CCC;
 207                                      END;
 208                                      ELSE DO;
 209                                              DO L=1 TO F;
 210                IF  (BLOK(L)=PAARL(1,K))|(BLOK(L)=PAARL(2,K)) THEN DO;
 211                         GEHAD(K)=1;
 212                          IF BLOK(L)= PAARL(1,K) THEN DO;
 213                                   DO O=1 TO F;
 214                         IF BLOK(O)= PAARL(2,K) THEN DO;
 215                         IF K<C THEN DO;
 216                         K=K+1;
 217                         GOTO AAA;
 218                         END;
 219                         ELSE GOTO CCC;
 220                         END;
 221                                   END;
 222                         BLOK(F+1)= PAARL(2,K);
 223                                                        END;
 224                                                               ELSE DO;
 225                                   DO O=1 TO F;
 226                         IF BLOK(O)= PAARL(1,K) THEN DO;
 227                         IF K<C THEN DO;
 228                         K=K+1;
```

```
 229                       GOTO AAA;
 230                       END;
 231                       ELSE GOTO CCC;
 232                       END;
 233                                 END;
 234                                 BLOK(F+1)= PAARL(1,K);
 235                                                                END;
 236                       F=F+1; K=D+1; GOTO AAA;
 239                                                                END;
 240                                            END;
 241                                  IF K<C THEN DO;
 242                                  K=K+1; GOTO AAA;
 244                                                END;
 245              ELSE GOTO CCC;
 246                                      END;
 247              CCC:
                  END;
 248                  DO L=D TO N;
 249                       IF GEHAD(L)¬=1 THEN DO;
 250                       DO O=1 TO F;
 251                      PARTRN(BLOK(O))=BLOKNO;
 252                       END;
 253                       D=L;
 254                        BLOKNO=BLOKNO+1;
 255                       GOTO BBB;
 256                       END;
 257                  END;
 258                  DO O=1 TO F;
 259                      PARTRN(BLOK(O))=BLOKNO;
 260                  END;
 261                 DO O=1 TO MR;
 262                 IF PARTRN(O)=00 THEN DO;
 263                      BLOKNO=BLOKNO+1;
 264                      PARTRN(O)=BLOKNO;
 265                           END;
 266                   END;
 267                 IF BLOKNO=1 THEN GOTO QQQ;
 268                 ELSE PARTLN(A)=1; PARTLN(B)=1;
 270                 L=1;
 271                 DO O=1 TO ML;
 272                 IF O=A|O=B THEN GOTO RRR;
 273                 ELSE DO; L=L+1; PARTLN(O)=L; END;
 277              RRR:    END;
                  /* TOEVOEGING VAN BASISGENERATOR AAN REEDS GEGENEREERDE PARTITIEPAREN
                  */

 278                 IF Z=0 THEN DO;
 279                 Z=1; DO O=1 TO MR;PARTR(1,O)=PARTRN(O); END;
 283                       DO O=1 TO ML; PARTL(1,O)=PARTLN(O); END;
 286                 GOTO QQQ;
 287                      END;
 288                 ELSE DO;
 289                 W=0;
 290                 DO  Y=1 TO Z;M=MR;
```

```
PARTP: PROCEDURE OPTICNS(MAIN);

 292            PL=ADDR(PARTL(Y,1)); PR=ADDR(PARTR(Y,1));
 294               DC C=1 TO M; PARTO(O)=PARTRPT(O); PARTN(O)=PARTRN(O); END;
 298               CALL CHEK; M=ML;
 300               IF V=2|V=3 THEN DO;
 301                    IF V=2 THEN W=2;
 302            DO O=1 TO M; PARTO(O)=PARTLPT(O);PARTN(O)=PARTLN(O); END;
 306                    CALL CHEK;
 307                    IF V=2|V=3 THEN GOTO JJJ;
 308                    IF V=1 THEN DO O=1 TO M; PARTLPT(O)=PARTN(O); END;
 311                    ELSE DO;
 312                    CALL OPTL;
 313                    DC O=1 TO M; PARTLPT(O)=PARTC(O); END;
 316                    END;
 317                    GOTO JJJ;
 318                         END;
 319                ELSE DO;
 320                IF V=4 THEN GOTO JJJ;
 321                    DO O=1 TO M; PARTO(C)=PARTLPT(O); PARTN(O)=PARTLN(O); END;
 325                    CALL CHEK;
 326                    IF V=1|V=2 THEN GOTO JJJ;
 327                          IF V=3 THEN DO O=1 TO M; PARTLN(O)=PARTO(O); END;
 330                         ELSE DO;
 331                          CALL OPTL;
 332                         DO O=1 TO M; PARTLN(O)= PARTC(O); END;
 335                              END;
 336                         GOTO JJJ;
 337                END;
 338            JJJ: END;
 339            IF W=2 THEN GOTO QQQ;
 340            Z=Z+1;
 341            PL=ADDR(PARTL(Z,1)); PR=ADDR(PARTR(Z,1));
 343            DC C=1 TC ML;PARTLPT(O)= PARTLN(O); END;
 346            DC C=1 TC MR;PARTRPT(O)= PARTRN(O); END;
 349               END;
 350            QQQ: END;
 351                    END;
 352            A=1; B=Z; CALL PRINT;
 355            ZZ=Z;

                /* GENEREER MET DE BASIS-MM-PAREN DE VOLGENDE GENERATIE */

 356            DC X=1 TC Z-1;
 357                                     PR=ADDR(PARTR(X,1));
 358               DC O=1 TO M;
 359               IF PARTRPT(O)>2 THEN GOTO OOO;
 360            END;
 361            GOTC PPP;
 362            OOO:
                    DO Y=X+1 TO Z;
 363                                     PR=ADDR(PARTR(Y,1));
 364                DO O=1 TO M;
 365                IF PARTRPT(O)>2 THEN GOTO SSS;
 366                END;
 367            GOTC TTT;
 368            SSS: DO O=1 TO M; PARTN(O)= PARTRPT(O); END;
```

PARTP: PROCEDURE OPTIONS(MAIN);

```
 371                                     PR=ADDR(PARTR(X,1));
 372                 DO O=1 TO M; PARTO(O)=PARTRPT(O); END;
 375                     CALL CHEK;
 376                     IF V¬=4 THEN GOTO TTT;
 377                     CALL OPTL;
 378                     DO O=1 TO M;
 379                     IF PARTC(O)>1 THEN GOTO UVW;
 380                     END;
 381                 GOTO TTT;
 382                 UVW: DO O=1 TO M; PARTN(O)=PARTC(O); PARTRN(O)=PARTC(O); END;
 386                     DO U=ZZ TO Z+1 BY -1;
 387                                     PR=ADDR(PARTR(U,1));
 388                     DO O=1 TO M; PARTO(O)= PARTRPT(O); END;
 391                     CALL CHEK;
 392                     IF V=2 THEN GOTO TTT;
 393                     END;
 394                     C=0;
 395                     DO U=1 TO Z;
 396                 PL=ADDR(PARTL(U,1)); PR=ADDR(PARTR(U,1));
 398                 M=MR;
 399                     DO O=1 TO M;PARTO(O)=PARTRPT(O); PARTN(O)=PARTRN(O); END;
 403                     CALL CHEK;
 404                     IF V=2 THEN GOTO TTT;
 405                 M=ML;
 406                     IF V¬=1 THEN GOTO UUU;
 407                     IF C=0 THEN DO;
 408                          DO O=1 TO M;PARTLN(O)=PARTLPT(O); END;
 411                          C=1;
 412                              GOTO UUU;
 413                              END;
 414                          DO O=1 TO M; PARTO(O)=PARTLPT(O); PARTN(O)= PARTLN(O); END;
 418                              CALL CHEK;
 419                     IF V=1|V=2 THEN GOTO UUU;
 420                     IF V=3 THEN DO ; DO O=1 TO M; PARTLN(O) =PARTLPT(O);END;
 424                                   GOTO UUU;
 425                              END;
 426                     CALL OPTL;
 427                     DO O=1 TO M; PARTLN(O)=PARTC(O); END;
 430                 UUU: END;
 431                    ZZ=ZZ+1;
 432                 PL=ADDR(PARTL(ZZ,1)); PR=ADDR(PARTR(ZZ,1));
 434                 DO O=1 TO ML; PARTLPT(O) =PARTLN(O); END;
 437                 DO O=1 TO MR; PARTRPT(O) =PARTRN(O); END;
 440                 M=MR;
 441                 TTT: END;
 442                 PPP: END;
 443                     IF ZZ=Z THEN GOTO EINDE;
 444                     A=Z+1;B=ZZ; CALL PRINT;

                    /* GENEREER ALLE VOLGENDE GENERATIES */

 447                 NNN:
                       DO X=1 TO Z;
 448                                     PR=ADDR(PARTR(X,1));
 449                    DO O=1 TO M;
```

PARTP: PROCEDURE OPTIONS(MAIN);

```
 450                 IF PARTRPT(O)>2 THEN GOTO VVV;
 451              END;
 452              GOTO WWW;
 453              VVV:
                      DO Y=A TO B;
 454                                   PR=ADDR(PARTR(Y,1));
 455                  DO O=1 TO M;
 456                  IF PARTRPT(O)>2 THEN GOTO XXX;
 457                  END;
 458              GOTO YYY;
 459              XXX: DO O=1 TO M; PARTN(O)= PARTRPT(O); END;
 462              PR=ADDR(PARTR(X,1));
 463              DO O=1 TO M; PARTO(O)=PARTRPT(O); END;
 466                  CALL CHEK;
 467                  IF V¬=4 THEN GOTO YYY;
 468                  CALL OPTL;
 469                  DO O=1 TO M;
 470                  IF PARTC(O)>1 THEN GOTO XYZ;
 471                  END;
 472              GOTO YYY;
 473              XYZ: DO O=1 TO M; PARTN(O)=PARTC(O); PARTRN(O)=PARTC(O); END;
 477                  DO U=ZZ TO Z+1 BY -1;
 478                                   PR=ADDR(PARTR(U,1));
 479                  DO O=1 TO M; PARTO(O)= PARTRPT(O); END;
 482                  CALL CHEK;
 483                  IF V=2 THEN GOTO YYY;
 484                  END;
 485                  C=0;
 486                  DO U=1 TO Z;
 487              PL=ADDR(PARTL(U,1)); PR=ADDR(PARTR(U,1));
 489              M=MR;
 490                  DO O=1 TO M;PARTO(O)=PARTRPT(O); PARTN(O)=PARTRN(O); END;
 494                  CALL CHEK;
 495                  IF V=2 THEN GOTO YYY;
 496              M=ML;
 497                  IF V¬=1 THEN GOTO ZZZ;
 498                  IF C=0 THEN DO;
 499                       DO O=1 TO M;PARTLN(O)=PARTLPT(O); END;
 502                       C=1;
 503                           GOTO ZZZ;
 504                           END;
 505                           DO O=1 TO M; PARTO(O)=PARTLPT(O); PARTN(O)=PARTLN(O); END;
 509                           CALL CHEK;
 510                  IF V=1|V=2 THEN GOTO ZZZ;
 511                  IF V=3 THEN DO ; DO O=1 TO M; PARTLN(O)=PARTLPT(O);END;
 515                                GOTO ZZZ;
 516                           END;
 517                  CALL OPTL;
 518                  DO O=1 TO M; PARTLN(O)=PARTC(O); END;
 521              ZZZ: END;
 522                 ZZ=ZZ+1;
 523              PL=ADDR(PARTL(ZZ,1)); PR=ADDR(PARTR(ZZ,1));
 525              DO O=1 TO ML; PARTLPT(O) =PARTLN(O); END;
 528              DO O=1 TO MR; PARTRPT(O) =PARTRN(O); END;
 531              M=MR;
```

```
PARTP: PROCEDURE OPTIONS(MAIN);

  532              YYY: END;
  533              WWW: END;
  534                  IF ZZ=B THEN GOTO EINDE;
  535                  A=B+1; B=ZZ; CALL PRINT; GOTO NNN;
  539              EINDE:
                   END PARTP;
```

// EXEC

```
 2    3
 4    5
 5    6
 1    1
 1    1
 1    1


VOLGENDE GENERATIE MM-PAREN

( 1- 2; 3; 4- 5- 6) ( 1; 2- 4; 3- 5; 6)
( 1- 3; 2; 4- 5- 6) ( 1; 2- 5; 3- 6; 4)
( 1- 4- 5- 6; 2; 3) ( 1- 2- 3; 4; 5; 6)
( 1; 2- 3; 4- 5- 6) ( 1; 2; 3; 4- 5- 6)
( 1; 2- 4- 5- 6; 3) ( 1- 4- 5; 2; 3; 6)
( 1; 2; 3- 4- 5- 6) ( 1- 5- 6; 2; 3; 4)
( 1; 2; 3; 4- 5- 6) ( 1; 2; 3; 4; 5; 6)

VOLGENDE GENERATIE MM-PAREN

( 1- 2- 3; 4- 5- 6) ( 1; 2- 3- 4- 5- 6)
( 1- 2- 4- 5- 6; 3) ( 1- 2- 3- 4- 5; 6)
( 1- 2; 3- 4- 5- 6) ( 1- 3- 5- 6; 2- 4)
( 1- 3- 4- 5- 6; 2) ( 1- 2- 3- 5- 6; 4)
( 1- 3; 2- 4- 5- 6) ( 1- 2- 4- 5; 3- 6)
( 1- 4- 5- 6; 2- 3) ( 1- 2- 3; 4- 5- 6)
( 1; 2- 3- 4- 5- 6) ( 1- 4- 5- 6; 2; 3)
```

EOJ PL/I 030 14•24•05,DURATION 00•17•19

MM paren voor de „moeilijke machine" 662

/ EXEC

17

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	4	6	8	10	12	14	16	2	4	6	8	10	12	14	16
3	6	9	12	15	2	5	8	11	14	1	4	7	10	13	16
4	8	12	16	4	8	12	16	4	8	12	16	4	8	12	16
5	10	15	4	9	14	3	8	13	2	7	12	1	6	11	16
6	12	2	8	14	4	10	16	6	12	2	8	14	4	10	16
7	14	5	12	3	10	1	8	15	6	13	4	11	2	9	16
8	16	8	16	8	16	8	16	8	16	8	16	8	16	8	16
9	2	11	4	13	6	15	8	1	10	3	12	5	14	7	16
10	4	14	8	2	12	6	16	10	4	14	8	2	12	6	16
11	6	1	12	7	2	13	8	3	14	9	4	15	10	5	16
12	8	4	16	12	8	4	16	12	8	4	16	12	8	4	16
13	10	7	4	1	14	11	8	5	2	15	12	9	6	3	16
14	12	10	8	6	4	2	16	14	12	10	8	6	4	2	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	16
16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16

OLGENDE GENERATIE MM-PAREN

```
1- 3- 9-11; 2- 6; 4-12; 5- 7-13-15; 8;10-14;16)  ( 1- 3- 9-11; 2- 6; 4-12; 5- 7-13-15; 8;10-14;16)
1- 5- 9-13; 2-10; 3- 7-11-15; 4; 6-14; 8;12;16)  ( 1- 5- 9-13; 2-10; 3- 7-11-15; 4; 6-14; 8;12;16)
1- 7; 2-14; 3- 5; 4-12; 6-10; 8; 9-15;11-13;16)  ( 1- 7; 2-14; 3- 5; 4-12; 6-10; 8; 9-15;11-13;16)
1- 9; 2; 3-11; 4; 5-13; 6; 7-15; 8;10;12;14;16)  ( 1- 9; 2; 3-11; 4; 5-13; 6; 7-15; 8;10;12;14;16)
1-15; 2-14; 3-13; 4-12; 5-11; 6-10; 7- 9; 8;16)  ( 1-15; 2-14; 3-13; 4-12; 5-11; 6-10; 7- 9; 8;16)
1; 2- 4- 6- 8-10-12-14-16; 3; 5; 7; 9;11;13;15)  ( 1; 2- 4- 6- 8-10-12-14-16; 3; 5; 7; 9;11;13;15)
1; 2- 6; 3; 4-12; 5; 7; 8; 9;10-14;11;13;15;16)  ( 1; 2- 6; 3; 4-12; 5; 7; 8; 9;10-14;11;13;15;16)
1; 2-10; 3; 4; 5; 6-14; 7; 8; 9;11;12;13;15;16)  ( 1; 2-10; 3; 4; 5; 6-14; 7; 8; 9;11;12;13;15;16)
1; 2-14; 3; 4-12; 5; 6-10; 7; 8; 9;11;13;15;16)  ( 1; 2-14; 3; 4-12; 5; 6-10; 7; 8; 9;11;13;15;16)
1; 2; 3; 4- 8-12-16; 5; 6; 7; 9;10;11;13;14;15)  ( 1; 2; 3; 4- 8-12-16; 5; 6; 7; 9;10;11;13;14;15)
1; 2; 3; 4-12; 5; 6; 7; 8; 9;10;11;13;14;15;16)  ( 1; 2; 3; 4-12; 5; 6; 7; 8; 9;10;11;13;14;15;16)
1; 2; 3; 4; 5; 6; 7; 8-16; 9;10;11;12;13;14;15)  ( 1; 2; 3; 4; 5; 6; 7; 8-16; 9;10;11;12;13;14;15)
```

OLGENDE GENERATIE MM-PAREN

```
1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4-12; 8;16)  ( 1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4-12; 8;16)
1- 3- 9-11; 2- 4- 6- 8-10-12-14-16; 5- 7-13-15)  ( 1- 3- 9-11; 2- 4- 6- 8-10-12-14-16; 5- 7-13-15)
1- 3- 9-11; 2- 6-10-14; 4-12; 5- 7-13-15; 8;16)  ( 1- 3- 9-11; 2- 6-10-14; 4-12; 5- 7-13-15; 8;16)
1- 3- 9-11; 2- 6; 4- 8-12-16; 5- 7-13-15;10-14)  ( 1- 3- 9-11; 2- 6; 4- 8-12-16; 5- 7-13-15;10-14)
1- 3- 9-11; 2- 6; 4-12; 5- 7-13-15; 8-16;10-14)  ( 1- 3- 9-11; 2- 6; 4-12; 5- 7-13-15; 8-16;10-14)
1- 5- 9-13; 2- 4- 6- 8-10-12-14-16; 3- 7-11-15)  ( 1- 5- 9-13; 2- 4- 6- 8-10-12-14-16; 3- 7-11-15)
1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4-12; 8;16)  ( 1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4-12; 8;16)
1- 5- 9-13; 2-10; 3- 7-11-15; 4- 8-12-16; 6-14)  ( 1- 5- 9-13; 2-10; 3- 7-11-15; 4- 8-12-16; 6-14)
1- 5- 9-13; 2-10; 3- 7-11-15; 4-12; 6-14; 8;16)  ( 1- 5- 9-13; 2-10; 3- 7-11-15; 4-12; 6-14; 8;16)
1- 5- 9-13; 2-10; 3- 7-11-15; 4; 6-14; 8-16;12)  ( 1- 5- 9-13; 2-10; 3- 7-11-15; 4; 6-14; 8-16;12)
1- 7- 9-15; 2-14; 3- 5-11-13; 4-12; 6-10; 8;16)  ( 1- 7- 9-15; 2-14; 3- 5-11-13; 4-12; 6-10; 8;16)
1- 7; 2- 4- 6- 8-10-12-14-16; 3- 5; 9-15;11-13)  ( 1- 7; 2- 4- 6- 8-10-12-14-16; 3- 5; 9-15;11-13)
1- 7; 2- 6-10-14; 3- 5; 4-12; 8; 9-15;11-13;16)  ( 1- 7; 2- 6-10-14; 3- 5; 4-12; 8; 9-15;11-13;16)
1- 7; 2-14; 3- 5; 4- 8-12-16; 6-10; 9-15;11-13)  ( 1- 7; 2-14; 3- 5; 4- 8-12-16; 6-10; 9-15;11-13)
1- 7; 2-14; 3- 5; 4-12; 6-10; 8-16; 9-15;11-13)  ( 1- 7; 2-14; 3- 5; 4-12; 6-10; 8-16; 9-15;11-13)
1- 9; 2- 4- 6- 8-10-12-14-16; 3-11; 5-13; 7-15)  ( 1- 9; 2- 4- 6- 8-10-12-14-16; 3-11; 5-13; 7-15)
1- 9; 2- 6; 3-11; 4-12; 5-13; 7-15; 8;10-14;16)  ( 1- 9; 2- 6; 3-11; 4-12; 5-13; 7-15; 8;10-14;16)
1- 9; 2-10; 3-11; 4; 5-13; 6-14; 7-15; 8;12;16)  ( 1- 9; 2-10; 3-11; 4; 5-13; 6-14; 7-15; 8;12;16)
1- 9; 2-14; 3-11; 4-12; 5-13; 6-10; 7-15; 8;16)  ( 1- 9; 2-14; 3-11; 4-12; 5-13; 6-10; 7-15; 8;16)
1- 9; 2; 3-11; 4- 8-12-16; 5-13; 6; 7-15;10;14)  ( 1- 9; 2; 3-11; 4- 8-12-16; 5-13; 6; 7-15;10;14)
1- 9; 2; 3-11; 4-12; 5-13; 6; 7-15; 8;10;14;16)  ( 1- 9; 2; 3-11; 4-12; 5-13; 6; 7-15; 8;10;14;16)
1- 9; 2; 3-11; 4; 5-13; 6; 7-15; 8-16;10;12;14)  ( 1- 9; 2; 3-11; 4; 5-13; 6; 7-15; 8-16;10;12;14)
1-15; 2- 4- 6- 8-10-12-14-16; 3-13; 5-11; 7- 9)  ( 1-15; 2- 4- 6- 8-10-12-14-16; 3-13; 5-11; 7- 9)
1-15; 2- 6-10-14; 3-13; 4-12; 5-11; 7- 9; 8;16)  ( 1-15; 2- 6-10-14; 3-13; 4-12; 5-11; 7- 9; 8;16)
1-15; 2-14; 3-13; 4- 8-12-16; 5-11; 6-10; 7- 9)  ( 1-15; 2-14; 3-13; 4- 8-12-16; 5-11; 6-10; 7- 9)
```

```
1-15; 2-14; 3-13; 4-12; 5-11; 6-10; 7- 9; 8-16) ( 1-15; 2-14; 3-13; 4-12; 5-11; 6-10; 7- 9; 8-16)
1; 2- 6-10-14; 3; 4-12; 5; 7; 8; 9;11;13;15;16) ( 1; 2- 6-10-14; 3; 4-12; 5; 7; 8; 9;11;13;15;16)
1; 2- 6; 3; 4- 8-12-16; 5; 7; 9;10-14;11;13;15) ( 1; 2- 6; 3; 4- 8-12-16; 5; 7; 9;10-14;11;13;15)
1; 2- 6; 3; 4-12; 5; 7; 8-16; 9;10-14;11;13;15) ( 1; 2- 6; 3; 4-12; 5; 7; 8-16; 9;10-14;11;13;15)
1; 2-10; 3; 4- 8-12-16; 5; 6-14; 7; 9;11;13;15) ( 1; 2-10; 3; 4- 8-12-16; 5; 6-14; 7; 9;11;13;15)
1; 2-10; 3; 4-12; 5; 6-14; 7; 8; 9;11;13;15;16) ( 1; 2-10; 3; 4-12; 5; 6-14; 7; 8; 9;11;13;15;16)
1; 2-10; 3; 4; 5; 6-14; 7; 8-16; 9;11;12;13;15) ( 1; 2-10; 3; 4; 5; 6-14; 7; 8-16; 9;11;12;13;15)
1; 2-14; 3; 4- 8-12-16; 5; 6-10; 7; 9;11;13;15) ( 1; 2-14; 3; 4- 8-12-16; 5; 6-10; 7; 9;11;13;15)
1; 2-14; 3; 4-12; 5; 6-10; 7; 8-16; 9;11;13;15) ( 1; 2-14; 3; 4-12; 5; 6-10; 7; 8-16; 9;11;13;15)
1; 2; 3; 4-12; 5; 6; 7; 8-16; 9;10;11;13;14;15) ( 1; 2; 3; 4-12; 5; 6; 7; 8-16; 9;10;11;13;14;15)

OLGENDE GENERATIE MM-PAREN

1- 3- 5- 7- 9-11-13-15; 2- 4- 6- 8-10-12-14-16) ( 1- 3- 5- 7- 9-11-13-15; 2- 4- 6- 8-10-12-14-16)
1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4- 8-12-16) ( 1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4- 8-12-16)
1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4-12; 8-16) ( 1- 3- 5- 7- 9-11-13-15; 2- 6-10-14; 4-12; 8-16)
1- 3- 9-11; 2- 6-10-14; 4- 8-12-16; 5- 7-13-15) ( 1- 3- 9-11; 2- 6-10-14; 4- 8-12-16; 5- 7-13-15)
1- 3- 9-11; 2- 6-10-14; 4-12; 5- 7-13-15; 8-16) ( 1- 3- 9-11; 2- 6-10-14; 4-12; 5- 7-13-15; 8-16)
1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4- 8-12-16) ( 1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4- 8-12-16)
1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4-12; 8-16) ( 1- 5- 9-13; 2- 6-10-14; 3- 7-11-15; 4-12; 8-16)
1- 5- 9-13; 2-10; 3- 7-11-15; 4-12; 6-14; 8-16) ( 1- 5- 9-13; 2-10; 3- 7-11-15; 4-12; 6-14; 8-16)
1- 7- 9-15; 2- 4- 6- 8-10-12-14-16; 3- 5-11-13) ( 1- 7- 9-15; 2- 4- 6- 8-10-12-14-16; 3- 5-11-13)
1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4-12; 8;16) ( 1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4-12; 8;16)
1- 7- 9-15; 2-14; 3- 5-11-13; 4- 8-12-16; 6-10) ( 1- 7- 9-15; 2-14; 3- 5-11-13; 4- 8-12-16; 6-10)
1- 7- 9-15; 2-14; 3- 5-11-13; 4-12; 6-10; 8-16) ( 1- 7- 9-15; 2-14; 3- 5-11-13; 4-12; 6-10; 8-16)
1- 7; 2- 6-10-14; 3- 5; 4- 8-12-16; 9-15;11-13) ( 1- 7; 2- 6-10-14; 3- 5; 4- 8-12-16; 9-15;11-13)
1- 7; 2- 6-10-14; 3- 5; 4-12; 8-16; 9-15;11-13) ( 1- 7; 2- 6-10-14; 3- 5; 4-12; 8-16; 9-15;11-13)
1- 9; 2- 6-10-14; 3-11; 4-12; 5-13; 7-15; 8;16) ( 1- 9; 2- 6-10-14; 3-11; 4-12; 5-13; 7-15; 8;16)
1- 9; 2- 6; 3-11; 4- 8-12-16; 5-13; 7-15;10-14) ( 1- 9; 2- 6; 3-11; 4- 8-12-16; 5-13; 7-15;10-14)
1- 9; 2- 6; 3-11; 4-12; 5-13; 7-15; 8-16;10-14) ( 1- 9; 2- 6; 3-11; 4-12; 5-13; 7-15; 8-16;10-14)
1- 9; 2-10; 3-11; 4- 8-12-16; 5-13; 6-14; 7-15) ( 1- 9; 2-10; 3-11; 4- 8-12-16; 5-13; 6-14; 7-15)
1- 9; 2-10; 3-11; 4-12; 5-13; 6-14; 7-15; 8;16) ( 1- 9; 2-10; 3-11; 4-12; 5-13; 6-14; 7-15; 8;16)
1- 9; 2-10; 3-11; 4; 5-13; 6-14; 7-15; 8-16;12) ( 1- 9; 2-10; 3-11; 4; 5-13; 6-14; 7-15; 8-16;12)
1- 9; 2-14; 3-11; 4- 8-12-16; 5-13; 6-10; 7-15) ( 1- 9; 2-14; 3-11; 4- 8-12-16; 5-13; 6-10; 7-15)
1- 9; 2-14; 3-11; 4-12; 5-13; 6-10; 7-15; 8-16) ( 1- 9; 2-14; 3-11; 4-12; 5-13; 6-10; 7-15; 8-16)
1- 9; 2; 3-11; 4-12; 5-13; 6; 7-15; 8-16;10;14) ( 1- 9; 2; 3-11; 4-12; 5-13; 6; 7-15; 8-16;10;14)
1-15; 2- 6-10-14; 3-13; 4- 8-12-16; 5-11; 7- 9) ( 1-15; 2- 6-10-14; 3-13; 4- 8-12-16; 5-11; 7- 9)
1-15; 2- 6-10-14; 3-13; 4-12; 5-11; 7- 9; 8-16) ( 1-15; 2- 6-10-14; 3-13; 4-12; 5-11; 7- 9; 8-16)
1; 2- 6-10-14; 3; 4- 8-12-16; 5; 7; 9;11;13;15) ( 1; 2- 6-10-14; 3; 4- 8-12-16; 5; 7; 9;11;13;15)
1; 2- 6-10-14; 3; 4-12; 5; 7; 8-16; 9;11;13;15) ( 1; 2- 6-10-14; 3; 4-12; 5; 7; 8-16; 9;11;13;15)
1; 2-10; 3; 4-12; 5; 6-14; 7; 8-16; 9;11;13;15) ( 1; 2-10; 3; 4-12; 5; 6-14; 7; 8-16; 9;11;13;15)

OLGENDE GENERATIE MM-PAREN

1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4- 8-12-16) ( 1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4- 8-12-16)
1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4-12; 8-16) ( 1- 7- 9-15; 2- 6-10-14; 3- 5-11-13; 4-12; 8-16)
1- 9; 2- 6-10-14; 3-11; 4- 8-12-16; 5-13; 7-15) ( 1- 9; 2- 6-10-14; 3-11; 4- 8-12-16; 5-13; 7-15)
1- 9; 2- 6-10-14; 3-11; 4-12; 5-13; 7-15; 8-16) ( 1- 9; 2- 6-10-14; 3-11; 4-12; 5-13; 7-15; 8-16)
1- 9; 2-10; 3-11; 4-12; 5-13; 6-14; 7-15; 8-16) ( 1- 9; 2-10; 3-11; 4-12; 5-13; 6-14; 7-15; 8-16)
```

// EXEC

```
 4    1    3
 5    2    4
 1    3    4
 2    3    6


OLGENDE GENERATIE MM-PAREN

 1- 2; 3; 4)  ( 1- 2; 3- 4- 5; 6)
 1- 3; 2; 4)  ( 1- 3- 4; 2; 5; 6)
 1- 4; 2; 3)  ( 1- 3- 6; 2- 4; 5)
 1; 2- 3; 4)  ( 1- 5; 2- 3; 4; 6)
 1; 2- 4; 3)  ( 1; 2- 3- 5; 4- 6)
 1; 2; 3- 4)  ( 1- 2; 3; 4- 6; 5)

OLGENDE GENERATIE MM-PAREN

 1- 2- 3; 4)  ( 1- 2- 3- 4- 5; 6)
 1- 2; 3- 4)  ( 1- 2; 3- 4- 5- 6)
 1- 3- 4; 2)  ( 1- 2- 3- 4- 6; 5)
 1; 2- 3- 4)  ( 1- 2- 3- 5; 4- 6)
```

// EXEC

```
1   5
1   5
2   6
2   6
3   7
3   7
4   8
4   8


VOLGENDE GENERATIE MM-PAREN

( 1- 2; 3- 4; 5- 6; 7- 8) ( 1; 2; 3; 4; 5; 6; 7; 8)
( 1- 2- 3- 4; 5- 6; 7- 8) ( 1- 2; 3; 4; 5- 6; 7; 8)
( 1- 2- 5- 6; 3- 4; 7- 8) ( 1- 3; 2; 4; 5- 7; 6; 8)
( 1- 2- 7- 8; 3- 4; 5- 6) ( 1- 4; 2; 3; 5- 8; 6; 7)
( 1- 2; 3- 4- 5- 6; 7- 8) ( 1; 2- 3; 4; 5; 6- 7; 8)
( 1- 2; 3- 4- 7- 8; 5- 6) ( 1; 2- 4; 3; 5; 6- 8; 7)
( 1- 2; 3- 4; 5- 6- 7- 8) ( 1; 2; 3- 4; 5; 6; 7- 8)

VOLGENDE GENERATIE MM-PAREN

( 1- 2- 3- 4- 5- 6; 7- 8) ( 1- 2- 3; 4; 5- 6- 7; 8)
( 1- 2- 3- 4- 7- 8; 5- 6) ( 1- 2- 4; 3; 5- 6- 8; 7)
( 1- 2- 3- 4; 5- 6- 7- 8) ( 1- 2; 3- 4; 5- 6; 7- 8)
( 1- 2- 5- 6- 7- 8; 3- 4) ( 1- 3- 4; 2; 5- 7- 8; 6)
( 1- 2- 5- 6; 3- 4- 7- 8) ( 1- 3; 2- 4; 5- 7; 6- 8)
( 1- 2- 7- 8; 3- 4- 5- 6) ( 1- 4; 2- 3; 5- 8; 6- 7)
( 1- 2; 3- 4- 5- 6- 7- 8) ( 1; 2- 3- 4; 5; 6- 7- 8)

VOLGENDE GENERATIE MM-PAREN

( 1- 2- 3- 4- 5- 6- 7- 8) ( 1- 2- 3- 4; 5- 6- 7- 8)
```

EOJ PL/I 021 11•55•26,DURATION 00•16•21

2• Berekening van M(s-s)(tau) en M(i-s)(tau)•

Het nu volgende programma berekent van alle tweebloksparttities tau alle bijbehorende M(s-s)(tau) en M(i-s)(tau)•

Het maximale aantal te verwerken toestanden (de gespecificeerde don't cares meegerekend) is 10; het aantal inputs mag maximaal 12 zijn•

Op de invoerkaarten moeten de gegevens van de machine als volgt worden geponst:

1) Eerste 2 plaatsen: het aantal toestanden zonder de gespecificeerde don't cares; bijvoorbeeld 04•
2) Derde en vierde plaats: het aantal toestanden inclusief de gespecificeerde don't cares; bijvoorbeeld 06•
3) Vijfde en zesde plaats: het aantal inputs; bijv 03•
4) Op de volgende plaatsen komt de inhoud van de volgorde tabel te staan in F(2)• De rijen worden van links naar rechts en van boven naar beneden uitgelezen•

28

```
' JOB PL/I,ECB,JANSENPG                    15/09/70 14-05-27 038  15•54•14
' OPTION LINK,LIST,SYM
' EXEC PL/I
```

MACHINE UIT WEINER 778

O P T I O N S   L I S T

PROCESS STMT

PTIONS TAKEN ARE 60C,LIST,SYM,ERRS,OPT,STMT•

```
TWOBLO:PROCEDURE OPTIONS(MAIN);

  1          TWOBLO:PROCEDURE OPTIONS(MAIN);
  2          DECLARE (A,B,C,D,E,F,G,H,I,J,K,ML,MR,N,P,X,Y,Z,HROW,HCOL,EEN,RNR,CNR,
                     MSVAR,MIVAR,MTVAR,STATEVAR,STATEDEC(12),TABLE(12,12))
                     FIXED BINARY(12),
                    (ROW(512,12),COL(512,12),REMAIN) BIT(12),
                     GEHAD(12) BIT(1),
                     ROWPT1(12) BIT(12) BASED(RPT1),
                     ROWPT2(12) BIT(12) BASED(RPT2),
                     ROWPT3(12) BIT(12) BASED(RPT3),
                     COLPT1(12) BIT(12) BASED(CPT1),
                     COLPT2(12) BIT(12) BASED(CPT2),
                     COLPT3(12) BIT(12) BASED(CPT3),
                    (RPT1,RPT2,RPT3,CPT1,CPT2,CPT3) POINTER;
  3          GET EDIT (ML,MR,N) (F(2));
  4          DO I=1 TO ML;DO J=1 TO N; GET EDIT (TABLE(I,J)) (F(2));
  7          PUT EDIT (TABLE(I,J)) (X(2),F(2)); END; PUT SKIP; END;
 11          K=0; C=FLOOR((MR-1)/3); D=(MR-1)-3*C;
 14          Z=1;
 15          DO A=MR TO 2 BY -1;
 16          DO I=1 TO ML;
 17          DO J=1 TO N;
 18            IF TABLE(I,J)=A THEN SUBSTR(ROW(Z,I),J,1)='1'B;
 19          END;
 20          END;
 21          DO I=1 TO N;
 22          DO J=1 TO ML;
 23            IF TABLE(J,I)=A THEN SUBSTR(COL(Z,I),J,1)='1'B;
 24          END;
 25          END;
 26          K=K+1;  Z=2**K;
 28          END;
 29          DO K=0 TO C-1;
 30          Z=8**K;
 31                  DO I=1 TO ML;
 32                  ROW(3*Z,I)=(ROW(2*Z,I)|ROW(Z,I));
 33                  ROW(5*Z,I)=(ROW(4*Z,I)|ROW(Z,I));
 34                  ROW(6*Z,I)=(ROW(4*Z,I)|ROW(2*Z,I));
 35                  ROW(7*Z,I)=(ROW(6*Z,I)|ROW(Z,I));
 36                  END;
 37                  DO I=1 TO N;
 38                  COL(3*Z,I)=(COL(2*Z,I)|COL(Z,I));
 39                  COL(5*Z,I)=(COL(4*Z,I)|COL(Z,I));
 40                  COL(6*Z,I)=(COL(4*Z,I)|COL(2*Z,I));
 41                  COL(7*Z,I)=(COL(6*Z,I)|COL(Z,I));
 42                  END;
 43             END;
 44          IF D=2 THEN DO;
 45             K=K+1; Z=8**K;
 47             DO I=1 TO ML;
 48             ROW(3*Z,I)=(ROW(2*Z,I)|ROW(Z,I));
 49             END;
 50             DO I=1 TO N;
 51             COL(3*Z,I)=(COL(2*Z,I)|COL(Z,I));
 52             END;
 53                  END;
```

```
WOBLO:PROCEDURE OPTIONS(MAIN);

  54           P=C-1; E=1; F=7;
  57           DDD:
               DO K=E TO P;
  58              DO H=1 TO F;
  59           Z=H*8**K;
  60              RPT1=ADDR(ROW(Z,1));
  61              CPT1=ADDR(COL(Z,1));
  62                     DO G=1 TO 8**K-1;
  63                     RPT2=ADDR(ROW(G,1));
  64                     CPT2=ADDR(COL(G,1));
  65                     RPT3=ADDR(ROW(Z+G,1));
  66                     CPT3=ADDR(COL(Z+G,1));
  67                      ROWPT3=(ROWPT2|ROWPT1);
  68                     COLPT3=(COLPT2|COLPT1);
  69                     END;
  70              END;
  71           END;
  72           IF D¬=0 THEN DO;
  73              IF F=7 THEN DO;
  74                     P=C; E=C; F=(2**D-1); GOTO DDD;
  78                     END;
  79              END;
  80           Z=Z+G-1;
  81           PUT SKIP EDIT ('  * = NIET TE GEBRUIKEN BIJ TOEWIJZING MET MINIMAAL AAN
               TAL VARIABELEN')(A);
  82           PUT SKIP EDIT ('ABS = AANTAL BLOKKEN IN MS-PARTITIE')(A);
  83           PUT SKIP EDIT ('AVS = MINIMALE AANTAL AFHANKELIJKE TOESTANDSVARIABELEN
               BIJ GESCHIKTE TOESTANDSTOEWIJZING')(A);
  84           PUT SKIP EDIT ('ABI = AANTAL BLOKKEN IN MI-PARTITIE')(A);
  85           PUT SKIP EDIT ('AVI = MINIMALE AANTAL AFHANKELIJKE INPUT VARIABELEN BIJ
                GESCHIKTE INPUT TOEWIJZING')(A);
  86           PUT SKIP EDIT ('TAV = MINIMALE AANTAL AFHANKELIJKE VARIABELEN BIJ GESCH
               IKTE TOESTANDS EN INPUT TOEWIJZING')(A);
  87           PUT SKIP (2);
  88           PUT SKIP EDIT ('TWEEBLOKS PARTITIE','MS-PARTITIE','MI-PARTITIE',
                              'ABS AVS ABI AVI TAV')(X(3),A(18),X(15),A(11),X(22),
                                                       A(11),X(21),A(19));
  89           DO I=1 TO Z;
  90           STATEDEC=1; Y=I; EEN=1; K=-1;
  94           AAA:
               K=K+1;
  95           IF Y¬=1 THEN DO;
  96                     X=MOD(Y,2);
  97                     IF X=1 THEN DO;
  98                               EEN =EEN+1;
  99                               STATEDEC(MR-K)=2;
 100                               END;
 101                     Y=FLOOR(Y/2);
 102                     GOTO AAA;
 103                     END;
 104           STATEDEC(MR-K)=2;
 105           STATEVAR=CEIL(LOG2(ML));
 106           K=2**(STATEVAR-1);
 107           IF MR¬=ML THEN DO;
 108                     EEN=1;
```

```
'WOBLO:PROCEDURE OPTIONS(MAIN);

 109                     DO J=2 TO ML;
 110                     IF STATEDEC(J)=2 THEN EEN=EEN+1;
 111             END;
 112                     END;
 113             IF (EEN>K)|(ML-EEN>K) THEN PUT SKIP EDIT ('* ( 1')(A(5));
 114                                  ELSE PUT SKIP EDIT ('  ( 1')(A(5));
 115             A=2; K=0; B=1;
 118             GGG:
                 DO J=A TO MR;
 119             IF STATEDEC(J)=B THEN PUT EDIT ('-',J)(A(1),F(2));
 120                            ELSE DO;
 121                            IF K¬=1 THEN DO;
 122                                    K=1; A=J;
 124                            END;
 125                     END;
 126             END;
 127             IF B=1 THEN DO;
 128                     PUT EDIT (';',A)(A(1),F(2));
 129                     B=2;
 130                     A=A+1;
 131                     GOTO GGG;
 132                     END;
 133             PUT EDIT (')')(A(1));
 134             DO J=1 TO (32-3*MR);
 135             PUT EDIT (' ')(A(1));
 136             END;
 137             PUT EDIT ('( 1')(A(3));
 138             A=2; GEHAD='0'B; GEHAD(1)='1'B; RNR=1;
 142             RPT1=ADDR(ROW(I,1)); REMAIN=ROWPT1(1);
 144             HHH:
                 DO J=A TO ML;
 145             IF ROWPT1(J)=REMAIN THEN DO;
 146                     PUT EDIT ('-',J)(A(1),F(2)); GEHAD(J)='1'B;
 148                     END;
 149             END;
 150             DO J=A TO ML;
 151             IF GEHAD(J)='0'B THEN DO;
 152                 RNR=RNR+1; REMAIN=ROWPT1(J); A=J+1; PUT EDIT (';',J)(A(1),F(2));
 156                 GOTO HHH;
 157                 END;
 158             END;
 159             PUT EDIT (')')(A(1));
 160             DO J=1 TO (32-3*ML);
 161             PUT EDIT (' ')(A(1));
 162             END;
 163             PUT EDIT ('( 1')(A(3));
 164             A=2; GEHAD='0'B; GEHAD(1)='1'B; CNR=1;
 168             CPT1=ADDR(COL(I,1)); REMAIN=COLPT1(1);
 170             III:
                 DO J=A TO N;
 171             IF COLPT1(J)=REMAIN THEN DO;
 172                     PUT EDIT ('-' ,J)(A(1),F(2)); GEHAD(J)='1'B;
 174                     END;
 175             END;
 176             DO J=A TO N;
```

```
WOBLO:PROCEDURE OPTIONS(MAIN);

 177                 IF GEHAD(J)='0'B THEN DO;
 178                        CNR=CNR+1; REMAIN=COLPT1(J); A=J+1;
 181                        PUT EDIT (';',J) (A(1),F(2)); GOTO III;
 183                        END;
 184                END;
 185                PUT EDIT (')') (A(1));
 186                DO J=1 TO (33-3*N);
 187                PUT EDIT (' ') (A(1));
 188                END;
 189                MSVAR=CEIL(LOG2(RNR)); MIVAR=CEIL(LOG2(CNR)); MTVAR=MSVAR+MIVAR;
 192                PUT EDIT (RNR,MSVAR,CNR,MIVAR,MTVAR) (F(2),X(2),F(2),X(2),F(2),X(2),
                                                     F(2),X(2),F(2));
 193                END;
 194                END TWOBLO;
```

1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	7	6	3	2	5	4	2
4	2	6	4	2	2	4	2
5	7	2	5	2	5	2	2
1	6	1	1	6	4	1	6
1	1	7	1	7	1	5	7

* = NIET TE GEBRUIKEN BIJ TOEWIJZING MET MINIMAAL AANTAL VARIABELEN
S = AANTAL BLOKKEN IN MS-PARTITIE
S = MINIMALE AANTAL AFHANKELIJKE TOESTANDSVARIABELEN BIJ GESCHIKTE TOESTANDSTOEWIJZING
I = AANTAL BLOKKEN IN MI-PARTITIE
I = MINIMALE AANTAL AFHANKELIJKE INPUT VARIABELEN BIJ GESCHIKTE INPUT TOEWIJZING
V = MINIMALE AANTAL AFHANKELIJKE VARIABELEN BIJ GESCHIKTE TOESTANDS EN INPUT TOEWIJZING

TWEEBLOKS PARTITIE	MS-PARTITIE	MI-PARTITIE	ABS	AVS	ABI	AVI	TAV
(1- 2- 3- 4- 5- 6; 7)	(1- 2- 4- 6; 3- 5; 7)	(1- 4- 6- 7; 2; 3- 5- 8)	3	2	3	2	4
(1- 2- 3- 4- 5- 7; 6)	(1- 2- 5- 7; 3- 4; 6)	(1- 4- 6- 7; 2- 5- 8; 3)	3	2	3	2	4
(1- 2- 3- 4- 5; 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4- 6- 7; 2; 3; 5- 8)	6	3	4	2	5
(1- 2- 3- 4- 6- 7; 5)	(1- 2- 4- 6; 3; 5; 7)	(1- 4; 2- 3- 5- 8; 6; 7)	4	2	4	2	4
(1- 2- 3- 4- 6; 5- 7)	(1- 2- 4- 6; 3; 5; 7)	(1- 4; 2- 6; 3- 5- 7- 8)	4	2	3	2	4
(1- 2- 3- 4- 7; 5- 6)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 5- 8; 3; 6; 7)	6	3	5	3	6
(1- 2- 3- 4; 5- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 2- 3- 5- 6- 7; 4)	(1- 2- 5- 7; 3; 4; 6)	(1- 4; 2- 3- 5- 8; 6; 7)	4	2	4	2	4
(1- 2- 3- 5- 6; 4- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3- 5- 8; 6; 7)	6	3	5	3	6
(1- 2- 3- 5- 7; 4- 6)	(1- 2- 5- 7; 3; 4; 6)	(1- 4; 2- 5- 6- 8; 3- 7)	4	2	3	2	4
(1- 2- 3- 5; 4- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 2- 3- 6- 7; 4- 5)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 3- 5- 8; 6; 7)	6	3	4	2	5
(1- 2- 3- 6; 4- 5- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3- 5- 8; 6; 7)	6	3	5	3	6
(1- 2- 3- 7; 4- 5- 6)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 5- 8; 3; 6; 7)	6	3	5	3	6
(1- 2- 3; 4- 5- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	6	3	4	2	5
(1- 2- 4- 5- 6- 7; 3)	(1- 2- 4- 5- 6- 7; 3)	(1- 4; 2- 3- 5- 6- 7- 8)	2	1	2	1	2
(1- 2- 4- 5- 6; 3- 7)	(1- 2- 4- 6; 3; 5; 7)	(1- 4; 2; 3- 5- 8; 6- 7)	4	2	4	2	4
(1- 2- 4- 5- 7; 3- 6)	(1- 2- 5- 7; 3; 4; 6)	(1- 4; 2- 5- 8; 3; 6- 7)	4	2	4	2	4
(1- 2- 4- 5; 3- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6- 7)	6	3	5	3	6
(1- 2- 4- 6- 7; 3- 5)	(1- 2- 4- 6; 3- 5; 7)	(1- 4- 6; 2- 3- 5- 8; 7)	3	2	3	2	4
(1- 2- 4- 6; 3- 5- 7)	(1- 2- 4- 6; 3- 5; 7)	(1- 2- 4- 6; 3- 5- 7- 8)	3	2	2	1	3
(1- 2- 4- 7; 3- 5- 6)	(1- 2; 3; 4; 5; 6; 7)	(1- 4- 6; 2- 5- 8; 3; 7)	6	3	4	2	5
(1- 2- 4; 3- 5- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4- 6; 2; 3; 5- 8; 7)	6	3	5	3	6
(1- 2- 5- 6- 7; 3- 4)	(1- 2- 5- 7; 3- 4; 6)	(1- 4- 7; 2- 3- 5- 8; 6)	3	2	3	2	4
(1- 2- 5- 6; 3- 4- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4- 7; 2; 3- 5- 8; 6)	6	3	4	2	5
(1- 2- 5- 7; 3- 4- 6)	(1- 2- 5- 7; 3- 4; 6)	(1- 3- 4- 7; 2- 5- 6- 8)	3	2	2	1	3
(1- 2- 5; 3- 4- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4- 7; 2; 3; 5- 8; 6)	6	3	5	3	6
(1- 2- 6- 7; 3- 4- 5)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 3- 5- 8; 6; 7)	6	3	4	2	5
(1- 2- 6; 3- 4- 5- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3- 5- 8; 6; 7)	6	3	5	3	6
(1- 2- 7; 3- 4- 5- 6)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 5- 8; 3; 6; 7)	6	3	5	3	6
(1- 2; 3- 4- 5- 6- 7)	(1- 2; 3; 4; 5; 6; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	6	3	4	2	5
(1- 3- 4- 5- 6- 7; 2)	(1- 6- 7; 2; 3; 4; 5)	(1- 4; 2- 6; 3- 7; 5- 8)	5	3	4	2	5
(1- 3- 4- 5- 6; 2- 7)	(1- 6; 2; 3; 4; 5; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 3- 4- 5- 7; 2- 6)	(1- 7; 2; 3; 4; 5; 6)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 3- 4- 5; 2- 6- 7)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	7	3	6	3	6
(1- 3- 4- 6- 7; 2- 5)	(1- 6; 2; 3; 4; 5; 7)	(1- 3- 4; 2; 5- 6- 8; 7)	6	3	4	2	5
(1- 3- 4- 6; 2- 5- 7)	(1- 6; 2- 5; 3- 4; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	4	2	4	2	4
(1- 3- 4- 7; 2- 5- 6)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3- 6; 5- 8; 7)	7	3	5	3	6
(1- 3- 4; 2- 5- 6- 7)	(1; 2- 5; 3- 4; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	5	3	6	3	6
(1- 3- 5- 6- 7; 2- 4)	(1- 7; 2; 3; 4; 5; 6)	(1- 2- 4; 3; 5- 7- 8; 6)	6	3	4	2	5
(1- 3- 5- 6; 2- 4- 7)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2- 7; 3; 5- 8; 6)	7	3	5	3	6
(1- 3- 5- 7; 2- 4- 6)	(1- 7; 2- 4; 3- 5; 6)	(1- 4; 2- 6; 3- 7; 5- 8)	4	2	4	2	4
(1- 3- 5; 2- 4- 6- 7)	(1; 2- 4; 3- 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	5	3	6	3	6

(1- 3- 6- 7; 2- 4- 5)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	7	3	6	3	6
(1- 3- 6; 2- 4- 5- 7)	(1; 2- 5; 3; 4; 6; 7)	(1- 4; 2; 3; 5- 7- 8; 6)	6	3	5	3	6
(1- 3- 7; 2- 4- 5- 6)	(1; 2- 4; 3; 5; 6; 7)	(1- 4; 2; 3; 5- 6- 8; 7)	6	3	5	3	6
(1- 3; 2- 4- 5- 6- 7)	(1; 2- 4- 5; 3; 6; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	5	3	4	2	5
(1- 4- 5- 6- 7; 2- 3)	(1- 6- 7; 2; 3; 4; 5)	(1- 4; 2- 6; 3- 7; 5- 8)	5	3	4	2	5
(1- 4- 5- 6; 2- 3- 7)	(1- 6; 2; 3; 4; 5; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 4- 5- 7; 2- 3- 6)	(1- 7; 2; 3; 4; 5; 6)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 4- 5; 2- 3- 6- 7)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	7	3	6	3	6
(1- 4- 6- 7; 2- 3- 5)	(1- 6; 2; 3; 4; 5; 7)	(1- 4; 2; 3; 5- 6- 8; 7)	6	3	5	3	6
(1- 4- 6; 2- 3- 5- 7)	(1- 6; 2- 5; 3; 4; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	5	3	4	2	5
(1- 4- 7; 2- 3- 5- 6)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2; 3- 6; 5- 8; 7)	7	3	5	3	6
(1- 4; 2- 3- 5- 6- 7)	(1; 2- 5; 3; 4; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 5- 6- 7; 2- 3- 4)	(1- 7; 2; 3; 4; 5; 6)	(1- 4; 2; 3; 5- 7- 8; 6)	6	3	5	3	6
(1- 5- 6; 2- 3- 4- 7)	(1; 2; 3; 4; 5; 6; 7)	(1- 4; 2- 7; 3; 5- 8; 6)	7	3	5	3	6
(1- 5- 7; 2- 3- 4- 6)	(1- 7; 2- 4; 3; 5; 6)	(1- 4; 2- 6; 3- 7; 5- 8)	5	3	4	2	5
(1- 5; 2- 3- 4- 6- 7)	(1; 2- 4; 3; 5; 6; 7)	(1- 4; 2; 3; 5- 8; 6; 7)	6	3	6	3	6
(1- 6- 7; 2- 3- 4- 5)	(1; 2; 3; 4; 5; 6; 7)	(1- 4- 5- 8; 2; 3; 6; 7)	7	3	5	3	6
(1- 6; 2- 3- 4- 5- 7)	(1; 2- 5; 3- 4; 6; 7)	(1- 2- 4; 3; 5- 7- 8; 6)	5	3	4	2	5
(1- 7; 2- 3- 4- 5- 6)	(1; 2- 4; 3- 5; 6; 7)	(1- 3- 4; 2; 5- 6- 8; 7)	5	3	4	2	5
(1; 2- 3- 4- 5- 6- 7)	(1; 2- 3- 4- 5; 6; 7)	(1- 4; 2- 6; 3- 7; 5- 8)	4	2	4	2	4

// EXEC

3	1	4	2
1	5	4	2
3	4	3	5
5	1	4	2
5	4	3	5

~~YNUMMERS~~
* = NIET TE GEBRUIKEN BIJ TOEWIJZING MET MINIMAAL AANTAL VARIABELEN
BS = AANTAL BLOKKEN IM MS-PARTITIE
VS = MINIMALE AANTAL AFHANKELIJKE TOESTANDSVARIABELEN BIJ GESCHIKTE TOESTANDSTOEWIJZING
BI = AANTAL BLOKKEN IN MI-PARTITIE
VI = MINIMALE AANTAL AFHANKELIJKE INPUT VARIABELEN BIJ GESCHIKTE INPUT TOEWIJZING
AV = MINIMALE AANTAL AFHANKELIJKE VARIABELEN BIJ GESCHIKTE TOESTANDS EN INPUT TOEWIJZING

TWEEBLOKS PARTITIE	MS-PARTITIE	MI-PARTITIE	ABS	AVS	ABI	AVI	TAV
(1- 2- 3- 4; 5)	(1; 2; 3; 4; 5)	(1; 2; 3; 4)	5	3	4	2	5
(1- 2- 3- 5; 4)	(1- 2- 4; 3- 5)	(1- 4; 2; 3)	2	1	3	2	3
(1- 2- 3; 4- 5)	(1; 2; 3; 4; 5)	(1; 2; 3; 4)	5	3	4	2	5
(1- 2- 4- 5; 3)	(1; 2- 4; 3; 5)	(1; 2- 4; 3)	4	2	3	2	4
(1- 2- 4; 3- 5)	(1- 4; 2; 3- 5)	(1; 2; 3- 4)	3	2	3	2	4
(1- 2- 5; 3- 4)	(1; 2- 4; 3; 5)	(1; 2; 3; 4)	4	2	4	2	4
(1- 2; 3- 4- 5)	(1- 4; 2; 3- 5)	(1; 2; 3; 4)	3	2	4	2	4
(1- 3- 4- 5; 2)	(1- 2- 4; 3- 5)	(1- 2- 3; 4)	2	1	2	1	2
(1- 3- 4; 2- 5)	(1- 3; 2; 4- 5)	(1; 2; 3; 4)	3	2	4	2	4
(1- 3- 5; 2- 4)	(1- 2- 4; 3- 5)	(1; 2; 3- 4)	2	1	3	2	3
(1- 3; 2- 4- 5)	(1; 2; 3; 4; 5)	(1; 2; 3; 4)	5	3	4	2	5
(1- 4- 5; 2- 3)	(1; 2- 4; 3; 5)	(1; 2; 3; 4)	4	2	4	2	4
(1- 4; 2- 3- 5)	(1- 4; 2; 3- 5)	(1; 2; 3; 4)	3	2	4	2	4
(1- 5; 2- 3- 4)	(1; 2- 4; 3; 5)	(1; 2; 3; 4)	4	2	4	2	4
(1; 2- 3- 4- 5)	(1- 4; 2; 3- 5)	(1; 2; 3- 4)	3	2	3	2	4

OJ PL/I 024 13•16•22,DURATION 00•07•05

EXEC

4	1	3
5	2	4
1	3	4
2	3	6

* = NIET TE GEBRUIKEN BIJ TOEWIJZING MET MINIMAAL AANTAL VARIABELEN
S = AANTAL BLOKKEN IN MS-PARTITIE
S = MINIMALE AANTAL AFHANKELIJKE TOESTANDSVARIABELEN BIJ GESCHIKTE TOESTANDSTOEWIJZING
I = AANTAL BLOKKEN IN MI-PARTITIE
I = MINIMALE AANTAL AFHANKELIJKE INPUT VARIABELEN BIJ GESCHIKTE INPUT TOEWIJZING
V = MINIMALE AANTAL AFHANKELIJKE VARIABELEN BIJ GESCHIKTE TOESTANDS EN INPUT TOEWIJZING

TWEEBLOKS PARTITIE	MS-PARTITIE	MI-PARTITIE	ABS	AVS	ABI	AVI	TAV
(1- 2- 3- 4- 5; 6)	(1- 2- 3; 4)	(1- 2; 3)	2	1	2	1	2
(1- 2- 3- 4- 6; 5)	(1- 3- 4; 2)	(1; 2- 3)	2	1	2	1	2
(1- 2- 3- 4; 5- 6)	(1- 3; 2; 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 3- 5- 6; 4)	(1; 2- 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 3- 5; 4- 6)	(1; 2- 3- 4)	(1; 2; 3)	2	1	3	2	3
(1- 2- 3- 6; 4- 5)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 2- 3; 4- 5- 6)	(1; 2; 3- 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 4- 5- 6; 3)	(1; 2; 3- 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 4- 5; 3- 6)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 2- 4- 6; 3- 5)	(1; 2; 3- 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 4; 3- 5- 6)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 2- 5- 6; 3- 4)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 2- 5; 3- 4- 6)	(1; 2; 3- 4)	(1; 2; 3)	3	2	3	2	4
(1- 2- 6; 3- 4- 5)	(1- 2; 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 2; 3- 4- 5- 6)	(1- 2; 3- 4)	(1; 2; 3)	2	1	3	2	3
(1- 3- 4- 5- 6; 2)	(1- 3; 2; 4)	(1; 2; 3)	3	2	3	2	4
(1- 3- 4- 5; 2- 6)	(1- 3; 2; 4)	(1- 3; 2)	3	2	2	1	3
(1- 3- 4- 6; 2- 5)	(1- 3; 2; 4)	(1; 2; 3)	3	2	3	2	4
(1- 3- 4; 2- 5- 6)	(1- 3; 2; 4)	(1; 2; 3)	3	2	3	2	4
(1- 3- 5- 6; 2- 4)	(1- 4; 2; 3)	(1; 2; 3)	3	2	3	2	4
(1- 3- 5; 2- 4- 6)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 3- 6; 2- 4- 5)	(1- 4; 2; 3)	(1; 2; 3)	3	2	3	2	4
(1- 3; 2- 4- 5- 6)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 4- 5- 6; 2- 3)	(1; 2- 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 4- 5; 2- 3- 6)	(1; 2- 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 4- 6; 2- 3- 5)	(1; 2- 4; 3)	(1; 2; 3)	3	2	3	2	4
(1- 4; 2- 3- 5- 6)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1- 5- 6; 2- 3- 4)	(1; 2- 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 5; 2- 3- 4- 6)	(1; 2- 3; 4)	(1; 2; 3)	3	2	3	2	4
(1- 6; 2- 3- 4- 5)	(1; 2; 3; 4)	(1; 2; 3)	4	2	3	2	4
(1; 2- 3- 4- 5- 6)	(1; 2- 4; 3)	(1; 2; 3)	3	2	3	2	4

2	9
3	1
4	2
5	3
6	4
7	5
8	6
9	7

* = NIET TE GEBRUIKEN BIJ TOEWIJZING MET MINIMAAL AANTAL VARIABELEN
S = AANTAL BLOKKEN IN MS-PARTITIE
S = MINIMALE AANTAL AFHANKELIJKE TOESTANDSVARIABELEN BIJ GESCHIKTE TOESTANDSTOEWIJZING
I = AANTAL BLOKKEN IN MI-PARTITIE
I = MINIMALE AANTAL AFHANKELIJKE INPUT VARIABELEN BIJ GESCHIKTE INPUT TOEWIJZING
V = MINIMALE AANTAL AFHANKELIJKE VARIABELEN BIJ GESCHIKTE TOESTANDS EN INPUT TOEWIJZING

TWEEBLOKS PARTITIE	MS-PARTITIE	MI-PARTITIE	ABS	AVS	ABI	AVI	TAV
(1- 2- 3- 4- 5- 6- 7- 8; 9)	(1; 2- 3- 4- 5- 6- 7; 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 6- 7- 9; 8)	(1- 2- 3- 4- 5- 6- 8; 7)	(1; 2)	2	1	2	1	2
(1- 2- 3- 4- 5- 6- 7; 8- 9)	(1; 2- 3- 4- 5- 6; 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 6- 8- 9; 7)	(1- 2- 3- 4- 5- 7; 6; 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 6- 8; 7- 9)	(1; 2- 3- 4- 5- 7; 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 5- 6- 9; 7- 8)	(1- 2- 3- 4- 5; 6- 7; 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 6; 7- 8- 9)	(1; 2- 3- 4- 5; 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 5- 7- 8- 9; 6)	(1- 2- 3- 4- 6- 8; 5; 7)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 7- 8; 6- 9)	(1- 7; 2- 3- 4- 6; 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 7- 9; 6- 8)	(1- 2- 3- 4- 6- 8; 5; 7)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 7; 6- 8- 9)	(1; 2- 3- 4- 6; 5- 8; 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 5- 8- 9; 6- 7)	(1- 2- 3- 4; 5- 6; 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 5- 8; 6- 7- 9)	(1- 7; 2- 3- 4; 5- 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 5- 9; 6- 7- 8)	(1- 2- 3- 4; 5- 6; 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 5; 6- 7- 8- 9)	(1; 2- 3- 4; 5- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 6- 7- 8- 9; 5)	(1- 2- 3- 5- 7- 8; 4; 6)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 6- 7- 8; 5- 9)	(1- 6; 2- 3- 5- 7; 4- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 6- 7- 9; 5- 8)	(1- 2- 3- 5- 8; 4- 7; 6)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 6- 7; 5- 8- 9)	(1- 6; 2- 3- 5; 4- 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 6- 8- 9; 5- 7)	(1- 2- 3- 5- 7; 4; 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 6- 8; 5- 7- 9)	(1; 2- 3- 5- 7; 4; 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 6- 9; 5- 7- 8)	(1- 2- 3- 5; 4- 7; 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 6; 5- 7- 8- 9)	(1; 2- 3- 5; 4- 7; 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 7- 8- 9; 5- 6)	(1- 2- 3- 8; 4- 5; 6- 7)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 7- 8; 5- 6- 9)	(1- 6- 7; 2- 3; 4- 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 4- 7- 9; 5- 6- 8)	(1- 2- 3- 8; 4- 5; 6; 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 7; 5- 6- 8- 9)	(1- 6; 2- 3; 4- 5- 8; 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 8- 9; 5- 6- 7)	(1- 2- 3; 4- 5; 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 8; 5- 6- 7- 9)	(1- 7; 2- 3; 4- 5; 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4- 9; 5- 6- 7- 8)	(1- 2- 3; 4- 5; 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 4; 5- 6- 7- 8- 9)	(1; 2- 3; 4- 5; 6- 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 6- 7- 8- 9; 4)	(1- 2- 4- 6- 7- 8; 3; 5)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6- 7- 8; 4- 9)	(1- 5; 2- 4- 6- 7; 3- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6- 7- 9; 4- 8)	(1- 2- 4- 6- 8; 3- 7; 5)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6- 7; 4- 8- 9)	(1- 5; 2- 4- 6; 3- 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6- 8- 9; 4- 7)	(1- 2- 4- 7; 3- 6; 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6- 8; 4- 7- 9)	(1- 5; 2- 4- 7; 3- 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 6- 9; 4- 7- 8)	(1- 2- 4; 3- 6- 7; 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 5- 6; 4- 7- 8- 9)	(1- 5; 2- 4; 3- 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 7- 8- 9; 4- 6)	(1- 2- 4- 6- 8; 3; 5; 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 7- 8; 4- 6- 9)	(1- 7; 2- 4- 6; 3- 8; 5)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 7- 9; 4- 6- 8)	(1- 2- 4- 6- 8; 3; 5- 7)	(1; 2)	3	2	2	1	3

(1- 2- 3- 5- 7; 4- 6- 8- 9)	(1; 2- 4- 6; 3- 8; 5- 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 8- 9; 4- 6- 7)	(1- 2- 4; 3- 6; 5; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 8; 4- 6- 7- 9)	(1- 7; 2- 4; 3- 6; 5- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5- 9; 4- 6- 7- 8)	(1- 2- 4; 3- 6; 5- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 5; 4- 6- 7- 8- 9)	(1; 2- 4; 3- 6; 5- 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 6- 7- 8- 9; 4- 5)	(1- 2- 7- 8; 3- 4; 5- 6)	(1; 2)	3	2	2	1	3
(1- 2- 3- 6- 7- 8; 4- 5- 9)	(1- 5- 6; 2- 7; 3- 4- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 6- 7- 9; 4- 5- 8)	(1- 2- 8; 3- 4- 7; 5- 6)	(1; 2)	3	2	2	1	3
(1- 2- 3- 6- 7; 4- 5- 8- 9)	(1- 5- 6; 2; 3- 4- 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 3- 6- 8- 9; 4- 5- 7)	(1- 2- 7; 3- 4; 5- 8; 6)	(1; 2)	4	2	2	1	3
(1- 2- 3- 6- 8; 4- 5- 7- 9)	(1- 5; 2- 7; 3- 4; 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 6- 9; 4- 5- 7- 8)	(1- 2; 3- 4- 7; 5- 8; 6)	(1; 2)	4	2	2	1	3
(1- 2- 3- 6; 4- 5- 7- 8- 9)	(1- 5; 2; 3- 4- 7; 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 7- 8- 9; 4- 5- 6)	(1- 2- 8; 3- 4; 5; 6- 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 7- 8; 4- 5- 6- 9)	(1- 6- 7; 2; 3- 4- 8; 5)	(1; 2)	4	2	2	1	3
(1- 2- 3- 7- 9; 4- 5- 6- 8)	(1- 2- 8; 3- 4; 5- 7; 6)	(1; 2)	4	2	2	1	3
(1- 2- 3- 7; 4- 5- 6- 8- 9)	(1- 6; 2; 3- 4- 8; 5- 7)	(1; 2)	4	2	2	1	3
(1- 2- 3- 8- 9; 4- 5- 6- 7)	(1- 2; 3- 4; 5- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 8; 4- 5- 6- 7- 9)	(1- 7; 2; 3- 4; 5- 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 3- 9; 4- 5- 6- 7- 8)	(1- 2; 3- 4; 5- 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 3; 4- 5- 6- 7- 8- 9)	(1; 2; 3- 4; 5- 6- 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 6- 7- 8- 9; 3)	(1- 3- 5- 6- 7- 8; 2; 4)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6- 7- 8; 3- 9)	(1- 4; 2- 8; 3- 5- 6- 7)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6- 7- 9; 3- 8)	(1- 3- 5- 6- 8; 2- 7; 4)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6- 7; 3- 8- 9)	(1- 4; 2- 7- 8; 3- 5- 6)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6- 8- 9; 3- 7)	(1- 3- 5- 7; 2- 6; 4- 8)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6- 8; 3- 7- 9)	(1- 4; 2- 6; 3- 5- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 6- 9; 3- 7- 8)	(1- 3- 5; 2- 6- 7; 4- 8)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 6; 3- 7- 8- 9)	(1- 4; 2- 6- 7; 3- 5; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 7- 8- 9; 3- 6)	(1- 3- 6- 8; 2- 5; 4- 7)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 7- 8; 3- 6- 9)	(1- 4- 7; 2- 5- 8; 3- 6)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 7- 9; 3- 6- 8)	(1- 3- 6- 8; 2- 5; 4; 7)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 7; 3- 6- 8- 9)	(1- 4; 2- 5- 8; 3- 6; 7)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 8- 9; 3- 6- 7)	(1- 3; 2- 5- 6; 4- 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 4- 5- 8; 3- 6- 7- 9)	(1- 4- 7; 2- 5- 6; 3; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5- 9; 3- 6- 7- 8)	(1- 3; 2- 5- 6; 4- 8; 7)	(1; 2)	4	2	2	1	3
(1- 2- 4- 5; 3- 6- 7- 8- 9)	(1- 4; 2- 5- 6; 3; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 7- 8- 9; 3- 5)	(1- 3- 5- 7- 8; 2; 4; 6)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 7- 8; 3- 5- 9)	(1- 6; 2- 8; 3- 5- 7; 4)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 7- 9; 3- 5- 8)	(1- 3- 5- 8; 2- 7; 4; 6)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 7; 3- 5- 8- 9)	(1- 6; 2- 7- 8; 3- 5; 4)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 8- 9; 3- 5- 7)	(1- 3- 5- 7; 2; 4- 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 8; 3- 5- 7- 9)	(1; 2; 3- 5- 7; 4- 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6- 9; 3- 5- 7- 8)	(1- 3- 5; 2- 7; 4- 6; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 6; 3- 5- 7- 8- 9)	(1; 2- 7; 3- 5; 4- 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 7- 8- 9; 3- 5- 6)	(1- 3- 8; 2- 5; 4; 6- 7)	(1; 2)	4	2	2	1	3
(1- 2- 4- 7- 8; 3- 5- 6- 9)	(1- 6- 7; 2- 5- 8; 3; 4)	(1; 2)	4	2	2	1	3
(1- 2- 4- 7- 9; 3- 5- 6- 8)	(1- 3- 8; 2- 5; 4- 7; 6)	(1; 2)	4	2	2	1	3
(1- 2- 4- 7; 3- 5- 6- 8- 9)	(1- 6; 2- 5- 8; 3; 4- 7)	(1; 2)	4	2	2	1	3
(1- 2- 4- 8- 9; 3- 5- 6- 7)	(1- 3; 2- 5; 4- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 8; 3- 5- 6- 7- 9)	(1- 7; 2- 5; 3; 4- 6- 8)	(1; 2)	4	2	2	1	3
(1- 2- 4- 9; 3- 5- 6- 7- 8)	(1- 3; 2- 5; 4- 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 4; 3- 5- 6- 7- 8- 9)	(1; 2- 5; 3; 4- 6- 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 5- 6- 7- 8- 9; 3- 4)	(1- 6- 7- 8; 2- 3; 4- 5)	(1; 2)	3	2	2	1	3
(1- 2- 5- 6- 7- 8; 3- 4- 9)	(1- 4- 5; 2- 3- 8; 6- 7)	(1; 2)	3	2	2	1	3
(1- 2- 5- 6- 7- 9; 3- 4- 8)	(1- 6- 8; 2- 3- 7; 4- 5)	(1; 2)	3	2	2	1	3
(1- 2- 5- 6- 7; 3- 4- 8- 9)	(1- 4- 5; 2- 3- 7- 8; 6)	(1; 2)	3	2	2	1	3
(1- 2- 5- 6- 8- 9; 3- 4- 7)	(1- 7; 2- 3- 6; 4- 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 5- 6- 8; 3- 4- 7- 9)	(1- 4- 5; 2- 3- 6; 7; 8)	(1; 2)	4	2	2	1	3
(1- 2- 5- 6- 9; 3- 4- 7- 8)	(1; 2- 3- 6- 7; 4- 5- 8)	(1; 2)	3	2	2	1	3

(1- 2- 5- 6; 3- 4- 7- 8- 9)	(1- 4- 5; 2- 3- 6- 7; 8)	(1; 2)	3	2	2	1	3
(1- 2- 5- 7- 8- 9; 3- 4- 6)	(1- 6- 8; 2- 3; 4- 7; 5)	(1; 2)	4	2	2	1	3
(1- 2- 5- 7- 8; 3- 4- 6- 9)	(1- 4- 7; 2- 3- 8; 5; 6)	(1; 2)	4	2	2	1	3
(1- 2- 5- 7- 9; 3- 4- 6- 8)	(1- 6- 8; 2- 3; 4; 5- 7)	(1; 2)	4	2	2	1	3
(1- 2- 5- 7; 3- 4- 6- 8- 9)	(1- 4; 2- 3- 8; 5- 7; 6)	(1; 2)	4	2	2	1	3
(1- 2- 5- 8- 9; 3- 4- 6- 7)	(1; 2- 3- 6; 4- 7- 8; 5)	(1; 2)	4	2	2	1	3
(1- 2- 5- 8; 3- 4- 6- 7- 9)	(1- 4- 7; 2- 3- 6; 5- 8)	(1; 2)	3	2	2	1	3
(1- 2- 5- 9; 3- 4- 6- 7- 8)	(1; 2- 3- 6; 4- 8; 5- 7)	(1; 2)	4	2	2	1	3
(1- 2- 5; 3- 4- 6- 7- 8- 9)	(1- 4; 2- 3- 6; 5- 7- 8)	(1; 2)	3	2	2	1	3
(1- 2- 6- 7- 8- 9; 3- 4- 5)	(1- 7- 8; 2- 3; 4; 5- 6)	(1; 2)	4	2	2	1	3
(1- 2- 6- 7- 8; 3- 4- 5- 9)	(1- 5- 6; 2- 3- 8; 4; 7)	(1; 2)	4	2	2	1	3
(1- 2- 6- 7- 9; 3- 4- 5- 8)	(1- 8; 2- 3- 7; 4; 5- 6)	(1; 2)	4	2	2	1	3
(1- 2- 6- 7; 3- 4- 5- 8- 9)	(1- 5- 6; 2- 3- 7- 8; 4)	(1; 2)	3	2	2	1	3
(1- 2- 6- 8- 9; 3- 4- 5- 7)	(1- 7; 2- 3; 4- 6; 5- 8)	(1; 2)	4	2	2	1	3
(1- 2- 6- 8; 3- 4- 5- 7- 9)	(1- 5; 2- 3; 4- 6- 8; 7)	(1; 2)	4	2	2	1	3
(1- 2- 6- 9; 3- 4- 5- 7- 8)	(1; 2- 3- 7; 4- 6; 5- 8)	(1; 2)	4	2	2	1	3
(1- 2- 6; 3- 4- 5- 7- 8- 9)	(1- 5; 2- 3- 7; 4- 6- 8)	(1; 2)	3	2	2	1	3
(1- 2- 7- 8- 9; 3- 4- 5- 6)	(1- 8; 2- 3; 4- 5; 6- 7)	(1; 2)	4	2	2	1	3
(1- 2- 7- 8; 3- 4- 5- 6- 9)	(1- 6- 7; 2- 3- 8; 4- 5)	(1; 2)	3	2	2	1	3
(1- 2- 7- 9; 3- 4- 5- 6- 8)	(1- 8; 2- 3; 4- 5- 7; 6)	(1; 2)	4	2	2	1	3
(1- 2- 7; 3- 4- 5- 6- 8- 9)	(1- 6; 2- 3- 8; 4- 5- 7)	(1; 2)	3	2	2	1	3
(1- 2- 8- 9; 3- 4- 5- 6- 7)	(1; 2- 3; 4- 5- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 2- 8; 3- 4- 5- 6- 7- 9)	(1- 7; 2- 3; 4- 5- 6- 8)	(1; 2)	3	2	2	1	3
(1- 2- 9; 3- 4- 5- 6- 7- 8)	(1; 2- 3; 4- 5- 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 2; 3- 4- 5- 6- 7- 8- 9)	(1; 2- 3; 4- 5- 6- 7- 8)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 6- 7- 8- 9; 2)	(1; 2- 4- 5- 6- 7- 8; 3)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 6- 7- 8; 2- 9)	(1; 2- 4- 5- 6- 7; 3; 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 6- 7- 9; 2- 8)	(1- 7; 2- 4- 5- 6- 8; 3)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 6- 7; 2- 8- 9)	(1; 2- 4- 5- 6; 3; 7- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 6- 8- 9; 2- 7)	(1- 6; 2- 4- 5- 7; 3- 8)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 6- 8; 2- 7- 9)	(1- 8; 2- 4- 5- 7; 3; 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 6- 9; 2- 7- 8)	(1- 6- 7; 2- 4- 5; 3- 8)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 6; 2- 7- 8- 9)	(1- 8; 2- 4- 5; 3; 6- 7)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 7- 8- 9; 2- 6)	(1- 5; 2- 4- 6- 8; 3- 7)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 7- 8; 2- 6- 9)	(1; 2- 4- 6; 3- 7; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 7- 9; 2- 6- 8)	(1- 5; 2- 4- 6- 8; 3; 7)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 7; 2- 6- 8- 9)	(1- 7; 2- 4- 6; 3; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 8- 9; 2- 6- 7)	(1- 5- 6; 2- 4; 3- 7- 8)	(1; 2)	3	2	2	1	3
(1- 3- 4- 5- 8; 2- 6- 7- 9)	(1- 8; 2- 4; 3- 7; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5- 9; 2- 6- 7- 8)	(1- 5- 6; 2- 4; 3- 8; 7)	(1; 2)	4	2	2	1	3
(1- 3- 4- 5; 2- 6- 7- 8- 9)	(1- 7- 8; 2- 4; 3; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6- 7- 8- 9; 2- 5)	(1- 4; 2- 5- 7- 8; 3- 6)	(1; 2)	3	2	2	1	3
(1- 3- 4- 6- 7- 8; 2- 5- 9)	(1; 2- 5- 7; 3- 6; 4- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6- 7- 9; 2- 5- 8)	(1- 4- 7; 2- 5- 8; 3- 6)	(1; 2)	3	2	2	1	3
(1- 3- 4- 6- 7; 2- 5- 8- 9)	(1; 2- 5; 3- 6; 4- 7- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6- 8- 9; 2- 5- 7)	(1- 4; 2- 5- 7; 3- 8; 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6- 8; 2- 5- 7- 9)	(1- 6- 8; 2- 5- 7; 3; 4)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6- 9; 2- 5- 7- 8)	(1- 4- 7; 2- 5; 3- 8; 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 6; 2- 5- 7- 8- 9)	(1- 6- 8; 2- 5; 3; 4- 7)	(1; 2)	4	2	2	1	3
(1- 3- 4- 7- 8- 9; 2- 5- 6)	(1- 4- 5; 2- 8; 3- 6- 7)	(1; 2)	3	2	2	1	3
(1- 3- 4- 7- 8; 2- 5- 6- 9)	(1; 2; 3- 6- 7; 4- 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 7- 9; 2- 5- 6- 8)	(1- 4- 5; 2- 8; 3- 6; 7)	(1; 2)	4	2	2	1	3
(1- 3- 4- 7; 2- 5- 6- 8- 9)	(1- 7; 2; 3- 6; 4- 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 4- 8- 9; 2- 5- 6- 7)	(1- 4- 5; 2; 3- 7- 8; 6)	(1; 2)	4	2	2	1	3
(1- 3- 4- 8; 2- 5- 6- 7- 9)	(1- 6- 8; 2; 3- 7; 4- 5)	(1; 2)	4	2	2	1	3
(1- 3- 4- 9; 2- 5- 6- 7- 8)	(1- 4- 5; 2; 3- 8; 6- 7)	(1; 2)	4	2	2	1	3
(1- 3- 4; 2- 5- 6- 7- 8- 9)	(1- 6- 7- 8; 2; 3; 4- 5)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 7- 8- 9; 2- 4)	(1; 2- 4- 6- 7- 8; 3; 5)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 7- 8; 2- 4- 9)	(1- 3; 2- 4- 6- 7; 5; 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 7- 9; 2- 4- 8)	(1- 7; 2- 4- 6- 8; 3; 5)	(1; 2)	4	2	2	1	3

(1- 3- 5- 6- 7; 2- 4- 8- 9)	(1- 3; 2- 4- 6; 5; 7- 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 8- 9; 2- 4- 7)	(1- 6; 2- 4- 7; 3; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 8; 2- 4- 7- 9)	(1- 3- 8; 2- 4- 7; 5; 6)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6- 9; 2- 4- 7- 8)	(1- 6- 7; 2- 4; 3; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 6; 2- 4- 7- 8- 9)	(1- 3- 8; 2- 4; 5; 6- 7)	(1; 2)	4	2	2	1	3
(1- 3- 5- 7- 8- 9; 2- 4- 6)	(1; 2- 4- 6- 8; 3- 5; 7)	(1; 2)	4	2	2	1	3
(1- 3- 5- 7- 8; 2- 4- 6- 9)	(1- 3- 5; 2- 4- 6; 7; 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 7- 9; 2- 4- 6- 8)	(1; 2- 4- 6- 8; 3- 5- 7)	(1; 2)	3	2	2	1	3
(1- 3- 5- 7; 2- 4- 6- 8- 9)	(1- 3- 5- 7; 2- 4- 6; 8)	(1; 2)	3	2	2	1	3
(1- 3- 5- 8- 9; 2- 4- 6- 7)	(1- 6; 2- 4; 3- 5; 7- 8)	(1; 2)	4	2	2	1	3
(1- 3- 5- 8; 2- 4- 6- 7- 9)	(1- 3- 5- 8; 2- 4; 6; 7)	(1; 2)	4	2	2	1	3
(1- 3- 5- 9; 2- 4- 6- 7- 8)	(1- 6; 2- 4; 3- 5- 7; 8)	(1; 2)	4	2	2	1	3
(1- 3- 5; 2- 4- 6- 7- 8- 9)	(1- 3- 5- 7- 8; 2- 4; 6)	(1; 2)	3	2	2	1	3
(1- 3- 6- 7- 8- 9; 2- 4- 5)	(1- 4; 2- 7- 8; 3; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 6- 7- 8; 2- 4- 5- 9)	(1- 3; 2- 7; 4- 8; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 6- 7- 9; 2- 4- 5- 8)	(1- 4- 7; 2- 8; 3; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 6- 7; 2- 4- 5- 8- 9)	(1- 3; 2; 4- 7- 8; 5- 6)	(1; 2)	4	2	2	1	3
(1- 3- 6- 8- 9; 2- 4- 5- 7)	(1- 4; 2- 7; 3- 6; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 6- 8; 2- 4- 5- 7- 9)	(1- 3- 6- 8; 2- 7; 4; 5)	(1; 2)	4	2	2	1	3
(1- 3- 6- 9; 2- 4- 5- 7- 8)	(1- 4- 7; 2; 3- 6; 5- 8)	(1; 2)	4	2	2	1	3
(1- 3- 6; 2- 4- 5- 7- 8- 9)	(1- 3- 6- 8; 2; 4- 7; 5)	(1; 2)	4	2	2	1	3
(1- 3- 7- 8- 9; 2- 4- 5- 6)	(1- 4; 2- 8; 3- 5; 6- 7)	(1; 2)	4	2	2	1	3
(1- 3- 7- 8; 2- 4- 5- 6- 9)	(1- 3- 5; 2; 4- 8; 6- 7)	(1; 2)	4	2	2	1	3
(1- 3- 7- 9; 2- 4- 5- 6- 8)	(1- 4; 2- 8; 3- 5- 7; 6)	(1; 2)	4	2	2	1	3
(1- 3- 7; 2- 4- 5- 6- 8- 9)	(1- 3- 5- 7; 2; 4- 8; 6)	(1; 2)	4	2	2	1	3
(1- 3- 8- 9; 2- 4- 5- 6- 7)	(1- 4; 2; 3- 5- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 3- 8; 2- 4- 5- 6- 7- 9)	(1- 3- 5- 6- 8; 2; 4; 7)	(1; 2)	4	2	2	1	3
(1- 3- 9; 2- 4- 5- 6- 7- 8)	(1- 4; 2; 3- 5- 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 3; 2- 4- 5- 6- 7- 8- 9)	(1- 3- 5- 6- 7- 8; 2; 4)	(1; 2)	3	2	2	1	3
(1- 4- 5- 6- 7- 8- 9; 2- 3)	(1- 2; 3- 4; 5- 6- 7- 8)	(1; 2)	3	2	2	1	3
(1- 4- 5- 6- 7- 8; 2- 3- 9)	(1; 2- 8; 3- 4; 5- 6- 7)	(1; 2)	4	2	2	1	3
(1- 4- 5- 6- 7- 9; 2- 3- 8)	(1- 2- 7; 3- 4; 5- 6- 8)	(1; 2)	3	2	2	1	3
(1- 4- 5- 6- 7; 2- 3- 8- 9)	(1; 2- 7- 8; 3- 4; 5- 6)	(1; 2)	4	2	2	1	3
(1- 4- 5- 6- 8- 9; 2- 3- 7)	(1- 2- 6; 3- 4- 8; 5- 7)	(1; 2)	3	2	2	1	3
(1- 4- 5- 6- 8; 2- 3- 7- 9)	(1- 8; 2- 6; 3- 4; 5- 7)	(1; 2)	4	2	2	1	3
(1- 4- 5- 6- 9; 2- 3- 7- 8)	(1- 2- 6- 7; 3- 4- 8; 5)	(1; 2)	3	2	2	1	3
(1- 4- 5- 6; 2- 3- 7- 8- 9)	(1- 8; 2- 6- 7; 3- 4; 5)	(1; 2)	4	2	2	1	3
(1- 4- 5- 7- 8- 9; 2- 3- 6)	(1- 2- 5; 3- 4- 7; 6- 8)	(1; 2)	3	2	2	1	3
(1- 4- 5- 7- 8; 2- 3- 6- 9)	(1; 2- 5- 8; 3- 4- 7; 6)	(1; 2)	4	2	2	1	3
(1- 4- 5- 7- 9; 2- 3- 6- 8)	(1- 2- 5; 3- 4; 6- 8; 7)	(1; 2)	4	2	2	1	3
(1- 4- 5- 7; 2- 3- 6- 8- 9)	(1- 7; 2- 5- 8; 3- 4; 6)	(1; 2)	4	2	2	1	3
(1- 4- 5- 8- 9; 2- 3- 6- 7)	(1- 2- 5- 6; 3- 4- 7- 8)	(1; 2)	2	1	2	1	2
(1- 4- 5- 8; 2- 3- 6- 7- 9)	(1- 8; 2- 5- 6; 3- 4- 7)	(1; 2)	3	2	2	1	3
(1- 4- 5- 9; 2- 3- 6- 7- 8)	(1- 2- 5- 6; 3- 4- 8; 7)	(1; 2)	3	2	2	1	3
(1- 4- 5; 2- 3- 6- 7- 8- 9)	(1- 7- 8; 2- 5- 6; 3- 4)	(1; 2)	3	2	2	1	3
(1- 4- 6- 7- 8- 9; 2- 3- 5)	(1- 2; 3- 6; 4; 5- 7- 8)	(1; 2)	4	2	2	1	3
(1- 4- 6- 7- 8; 2- 3- 5- 9)	(1- 4; 2- 8; 3- 6; 5- 7)	(1; 2)	4	2	2	1	3
(1- 4- 6- 7- 9; 2- 3- 5- 8)	(1- 2- 7; 3- 6; 4; 5- 8)	(1; 2)	4	2	2	1	3
(1- 4- 6- 7; 2- 3- 5- 8- 9)	(1- 4; 2- 7- 8; 3- 6; 5)	(1; 2)	4	2	2	1	3
(1- 4- 6- 8- 9; 2- 3- 5- 7)	(1- 2; 3- 8; 4- 6; 5- 7)	(1; 2)	4	2	2	1	3
(1- 4- 6- 8; 2- 3- 5- 7- 9)	(1- 4- 6- 8; 2; 3; 5- 7)	(1; 2)	4	2	2	1	3
(1- 4- 6- 9; 2- 3- 5- 7- 8)	(1- 2- 7; 3- 8; 4- 6; 5)	(1; 2)	4	2	2	1	3
(1- 4- 6; 2- 3- 5- 7- 8- 9)	(1- 4- 6- 8; 2- 7; 3; 5)	(1; 2)	4	2	2	1	3
(1- 4- 7- 8- 9; 2- 3- 5- 6)	(1- 2- 5; 3- 6- 7; 4; 8)	(1; 2)	4	2	2	1	3
(1- 4- 7- 8; 2- 3- 5- 6- 9)	(1- 4; 2- 5- 8; 3- 6- 7)	(1; 2)	3	2	2	1	3
(1- 4- 7- 9; 2- 3- 5- 6- 8)	(1- 2- 5; 3- 6; 4- 7; 8)	(1; 2)	4	2	2	1	3
(1- 4- 7; 2- 3- 5- 6- 8- 9)	(1- 4- 7; 2- 5- 8; 3- 6)	(1; 2)	3	2	2	1	3
(1- 4- 8- 9; 2- 3- 5- 6- 7)	(1- 2- 5; 3- 7- 8; 4- 6)	(1; 2)	3	2	2	1	3
(1- 4- 8; 2- 3- 5- 6- 7- 9)	(1- 4- 6- 8; 2- 5; 3- 7)	(1; 2)	3	2	2	1	3
(1- 4- 9; 2- 3- 5- 6- 7- 8)	(1- 2- 5; 3- 8; 4- 6- 7)	(1; 2)	3	2	2	1	3

(1- 4; 2- 3- 5- 6- 7- 8- 9)	(1- 4- 6- 7- 8; 2- 5; 3)	(1; 2)	3	2	2	1	3
(1- 5- 6- 7- 8- 9; 2- 3- 4)	(1- 2; 3; 4- 5; 6- 7- 8)	(1; 2)	4	2	2	1	3
(1- 5- 6- 7- 8; 2- 3- 4- 9)	(1- 3; 2- 8; 4- 5; 6- 7)	(1; 2)	4	2	2	1	3
(1- 5- 6- 7- 9; 2- 3- 4- 8)	(1- 2- 7; 3; 4- 5; 6- 8)	(1; 2)	4	2	2	1	3
(1- 5- 6- 7; 2- 3- 4- 8- 9)	(1- 3; 2- 7- 8; 4- 5; 6)	(1; 2)	4	2	2	1	3
(1- 5- 6- 8- 9; 2- 3- 4- 7)	(1- 2- 6; 3; 4- 5- 8; 7)	(1; 2)	4	2	2	1	3
(1- 5- 6- 8; 2- 3- 4- 7- 9)	(1- 3- 8; 2- 6; 4- 5; 7)	(1; 2)	4	2	2	1	3
(1- 5- 6- 9; 2- 3- 4- 7- 8)	(1- 2- 6- 7; 3; 4- 5- 8)	(1; 2)	3	2	2	1	3
(1- 5- 6; 2- 3- 4- 7- 8- 9)	(1- 3- 8; 2- 6- 7; 4- 5)	(1; 2)	3	2	2	1	3
(1- 5- 7- 8- 9; 2- 3- 4- 6)	(1- 2; 3- 5; 4- 7; 6- 8)	(1; 2)	4	2	2	1	3
(1- 5- 7- 8; 2- 3- 4- 6- 9)	(1- 3- 5; 2- 8; 4- 7; 6)	(1; 2)	4	2	2	1	3
(1- 5- 7- 9; 2- 3- 4- 6- 8)	(1- 2; 3- 5- 7; 4; 6- 8)	(1; 2)	4	2	2	1	3
(1- 5- 7; 2- 3- 4- 6- 8- 9)	(1- 3- 5- 7; 2- 8; 4; 6)	(1; 2)	4	2	2	1	3
(1- 5- 8- 9; 2- 3- 4- 6- 7)	(1- 2- 6; 3- 5; 4- 7- 8)	(1; 2)	3	2	2	1	3
(1- 5- 8; 2- 3- 4- 6- 7- 9)	(1- 3- 5- 8; 2- 6; 4- 7)	(1; 2)	3	2	2	1	3
(1- 5- 9; 2- 3- 4- 6- 7- 8)	(1- 2- 6; 3- 5- 7; 4- 8)	(1; 2)	3	2	2	1	3
(1- 5; 2- 3- 4- 6- 7- 8- 9)	(1- 3- 5- 7- 8; 2- 6; 4)	(1; 2)	3	2	2	1	3
(1- 6- 7- 8- 9; 2- 3- 4- 5)	(1- 2; 3- 4; 5- 6; 7- 8)	(1; 2)	4	2	2	1	3
(1- 6- 7- 8; 2- 3- 4- 5- 9)	(1- 3- 4; 2- 8; 5- 6; 7)	(1; 2)	4	2	2	1	3
(1- 6- 7- 9; 2- 3- 4- 5- 8)	(1- 2- 7; 3- 4; 5- 6; 8)	(1; 2)	4	2	2	1	3
(1- 6- 7; 2- 3- 4- 5- 8- 9)	(1- 3- 4; 2- 7- 8; 5- 6)	(1; 2)	3	2	2	1	3
(1- 6- 8- 9; 2- 3- 4- 5- 7)	(1- 2; 3- 4- 6; 5- 8; 7)	(1; 2)	4	2	2	1	3
(1- 6- 8; 2- 3- 4- 5- 7- 9)	(1- 3- 4- 6- 8; 2; 5; 7)	(1; 2)	4	2	2	1	3
(1- 6- 9; 2- 3- 4- 5- 7- 8)	(1- 2- 7; 3- 4- 6; 5- 8)	(1; 2)	3	2	2	1	3
(1- 6; 2- 3- 4- 5- 7- 8- 9)	(1- 3- 4- 6- 8; 2- 7; 5)	(1; 2)	3	2	2	1	3
(1- 7- 8- 9; 2- 3- 4- 5- 6)	(1- 2; 3- 4- 5; 6- 7; 8)	(1; 2)	4	2	2	1	3
(1- 7- 8; 2- 3- 4- 5- 6- 9)	(1- 3- 4- 5; 2- 8; 6- 7)	(1; 2)	3	2	2	1	3
(1- 7- 9; 2- 3- 4- 5- 6- 8)	(1- 2; 3- 4- 5- 7; 6; 8)	(1; 2)	4	2	2	1	3
(1- 7; 2- 3- 4- 5- 6- 8- 9)	(1- 3- 4- 5- 7; 2- 8; 6)	(1; 2)	3	2	2	1	3
(1- 8- 9; 2- 3- 4- 5- 6- 7)	(1- 2; 3- 4- 5- 6; 7- 8)	(1; 2)	3	2	2	1	3
(1- 8; 2- 3- 4- 5- 6- 7- 9)	(1- 3- 4- 5- 6- 8; 2; 7)	(1; 2)	3	2	2	1	3
(1- 9; 2- 3- 4- 5- 6- 7- 8)	(1- 2; 3- 4- 5- 6- 7; 8)	(1; 2)	3	2	2	1	3
(1; 2- 3- 4- 5- 6- 7- 8- 9)	(1- 3- 4- 5- 6- 7- 8; 2)	(1; 2)	2	1	2	1	2

J PL/I 010 13•25•06,DURATION 00•10•15

3• Selektie in delta-rijen•

Het nu volgende programma selekteert de toekenningen naar het minimale aantal bijbehorende afhankelijkheden (hetgeen niet betekent dat de machine met deze afhankelijkheden gerealiseerd kan worden)•
Het programma vergt een grote geheugenruimte en is daarom wat minder geschikt•
Het maximale aantal te verwerken toestanden en inputs is dan ook slechts 8•

De volgorde tabel wordt ingelezen als beschreven in hoofdstuk 2)•

```
JOB PL/I,ECE,JANSENPG
OPTION LINK,LIST,SYM
EXEC PL/I
```

17/09/70 11-43-08 018 12•27•28

TWOBLO MET DELTARYEN. 554

53

O P T I O N S L I S T

PROCESS STMT

TIONS TAKEN ARE 60C,LIST,SYM,ERRS,OPT,STMT•

```
WOBLO:PROCEDURE OPTIONS(MAIN);

  1          TWOBLO:PROCEDURE OPTIONS(MAIN);
  2          DECLARE (A,C,D,E,F,G,H,I,J,K,M,N,P,Z,HROW,HCOL) FIXED BINARY(8),
             (DELTANR(6),EEN,STATEVAR, MIVAR,RNR,CNR,B,X,Y) FIXED BINARY(8),
                       TABLE(8,8) FIXED BINARY(4),
             (DELTAST(6,128,8),DELTAMM(6,128,8),DELTAII(6,128,8),DELTANRS(6,128),
             DELTANRI(6,128),STDEC(8),REMRY(8),REMCOL(8)) FIXED DECIMAL(1),
             REMAIN BIT(8),
                      ROW(256,8) BIT(8),
                      COL(256,8) BIT(8),
                      ROWPT1(8) BIT(8) BASED(RPT1),
                      COLPT1(8) BIT(8) BASED(CPT1),
                      ROWPT2(8) BIT(8) BASED(RPT2),
                      COLPT2(8) BIT(8) BASED(CPT2),
                      ROWPT3(8) BIT(8) BASED(RPT3),
                      COLPT3(8) BIT(8) BASED(CPT3),
                      (CPT1,CPT2,CPT3) POINTER,
                      (RPT1,RPT2,RPT3) POINTER;

  3          PRINT: PROCEDURE;
  4          E=16-2*M; F=16-2*N;
  6          DO A=1 TO  (STATEVAR+CEIL(LOG2(N)));
  7          PUT SKIP;
  8          PUT SKIP;
  9          PUT SKIP  EDIT ('AANTAL VARIABELEN=',A) (A(18),F(2));
 10          PUT SKIP EDIT ('TWEEBLOKPARTITIE','MSPARTITIE','MIPARTITIE',
             'BLCKNR S','BLCKNR I') (A(16),X(4),A(10),X(10),A(10),X(10),A(8),X(4),
             A(8));
 11          DO B=1 TO DELTANR(A);
 12          PUT SKIP EDIT ((' 'DO I=1 TO E)) (A(1));
 13          DO C=1 TO M;
 14          PUT EDIT (DELTAST(A,B,C)) (F(2));
 15          END;
 16          PUT EDIT ((' 'DO I=1 TO E)) (A(1));
 17          DO C=1 TO M;
 18          PUT EDIT (DELTAMM(A,B,C)) (F(2));
 19          END;
 20          PUT EDIT ((' 'DO I=1 TO F)) (A(1));
 21          DO C=1 TO N;
 22          PUT EDIT (DELTAII(A,B,C)) (F(2));
 23          END;
 24          PUT EDIT (DELTANRS(A,B),DELTANRI(A,B)) (X(11),F(2),X(10),F(2));END;END;
 27          END PRINT;

 28          GET EDIT (M,N) (F(1));
 29          DO I=1 TO M; DO J=1 TO N; GET EDIT (TABLE(I,J)) (F(1));
 32          PUT EDIT (TABLE(I,J)) (X(2),F(1)); END; PUT SKIP; END;
 36          HROW=0; ROW=0; C=FLOOR((M-1)/3); D=(M-1)-3*C; K=0; COL=0; HCOL=0;
 43          Z=1;
 44          DO A=M TO 2 BY -1;
 45          DO I=1 TO M;
 46          DO J=1 TO N;
 47            IF TABLE(I,J)=A THEN SUBSTR(ROW(Z,I),J,1)='1'B;
 48          END;
 49          END;
 50          DO I=1 TO N;
```

```
WOBLO:PROCEDURE OPTIONS(MAIN);

 51              DO J=1 TO M;
 52                IF TABLE(J,I)=A THEN SUBSTR(COL(Z,I),J,1)='1'B;
 53              END;
 54              END;
 55              K=K+1;  Z=2**K;
 57              END;
 58              DO K=0 TO C-1;
 59              Z=8**K;
 60                      DO I=1 TO M;
 61                      ROW(3*Z,I)=(ROW(2*Z,I)|ROW(Z,I));
 62                      ROW(5*Z,I)=(ROW(4*Z,I)|ROW(Z,I));
 63                      ROW(6*Z,I)=(ROW(4*Z,I)|ROW(2*Z,I));
 64                      ROW(7*Z,I)=(ROW(6*Z,I)|ROW(Z,I));
 65                      END;
 66                      DO I=1 TO N;
 67                      COL(3*Z,I)=(COL(2*Z,I)|COL(Z,I));
 68                      COL(5*Z,I)=(COL(4*Z,I)|COL(Z,I));
 69                      COL(6*Z,I)=(COL(4*Z,I)|COL(2*Z,I));
 70                      COL(7*Z,I)=(COL(6*Z,I)|COL(Z,I));
 71                      END;
 72                 END;
 73              IF D=2 THEN DO;
 74                 Z=8**K;
 75                 DO I=1 TO M;
 76                 ROW(3*Z,I)=(ROW(2*Z,I)|ROW(Z,I));
 77                 END;
 78                 DO I=1 TO N;
 79                 COL(3*Z,I)=(COL(2*Z,I)|COL(Z,I));
 80                 END;
 81                      END;
 82              P=C-1; E=1; F=7;
 85              DDD:
                 DO K=E TO P;
 86                 DO H=1 TO F;
 87              Z=H*8**K;
 88                 RPT1=ADDR(ROW(Z,1));
 89                 CPT1=ADDR(COL(Z,1));
 90                      DO G=1 TO 8**K-1;
 91                      RPT2=ADDR(ROW(G,1));
 92                      CPT2=ADDR(COL(G,1));
 93                      RPT3=ADDR(ROW(Z+G,1));
 94                      CPT3=ADDR(COL(Z+G,1));
 95                       ROWPT3=(ROWPT2|ROWPT1);
 96                      COLPT3=(COLPT2|COLPT1);
 97                      END;
 98                 END;
 99              END;
100              IF D¬=0 THEN DO;
101                 IF F=7 THEN DO;
102                      P=C; E=C; F=(2**D-1); GOTO DDD;
106                      END;
107                 END;
108              Z=Z+G-1;
109              DELTANB=0;
110              DO I=1 TO Z;
```

```
'WOBLO:PROCEDURE OPTIONS(MAIN);

 111              STDEC =1; Y=I; EEN=1; K=-1;
 115              AAA:
                  K=K+1;
 116              IF Y¬=1 THEN DO;
 117                      X=MOD(Y,2);
 118                      IF X=1 THEN DO;
 119                           EEN=EEN+1;
 120                           STDEC(M-K)=2;
 121                           END;
 122                      Y=FLOOR(Y/2);
 123                      GOTO AAA;
 124                      END;
 125                      STDEC(M-K)=2;
 126                      STATEVAR=CEIL(LOG2(M));
 127                      K=2**(STATEVAR-1);
 128              IF  (EEN>K)|((M-EEN)>K) THEN GOTO EEE;
 129              B=1; REMRY=0; REMRY(1)=1; A=2; RNR=1;
 134              RPT1=ADDR(ROW(I,1));
 135              EEE:
                  REMAIN=ROWPT1(B);
 136              DO J=A TO M;
 137              IF ROWPT1(J)=REMAIN THEN REMRY(J)=RNR;
 138              END;
 139              DO J=A TO M;
 140              IF REMRY(J)=0 THEN DO;
 141                      B=J; A=B+1; RNR=RNR+1; REMRY(B)=RNR; GOTO EEE;
 146                      END; END;
 148              B=1; REMCOL=0; REMCOL(1)=1; A=2; CNR=1;
 153              CPT1=ADDR(COL(I,1));
 154              FFF:
                  REMAIN=COLPT1(B);
 155              DO J=A TO N;
 156              IF COLPT1(J)=REMAIN THEN REMCOL(J)=CNR;
 157              END;
 158              DO J=A TO N;
 159              IF REMCOL(J)=0 THEN DO;
 160                      B=J; A=B+1; CNR=CNR+1; REMCOL(B)=CNR;GOTO FFF;
 165                      END; END;
 167              MIVAR=CEIL(LOG2(RNR))+CEIL(LOG2(CNR));
 168              DELTANR(MIVAR)=DELTANR(MIVAR)+1;
 169              DO J=1 TO M;
 170              DELTAST(MIVAR,DELTANR(MIVAR),J)=STDEC(J);
 171              DELTAMM(MIVAR,DELTANR(MIVAR),J)=REMRY(J);
 172              END;
 173              DO J=1 TO N;
 174              DELTAII(MIVAR,DELTANR(MIVAR),J)=REMCOL(J);
 175              END;
 176              DELTANRS(MIVAR,DELTANR(MIVAR))=RNR;DELTANRI(MIVAR,DELTANR(MIVAR))=CNR;
 178              EEE: END;
 179              CALL PRINT;
 180              END TWOBLO;
```

```
3  1  4  2
1  5  4  2
3  4  3  5
5  1  4  2
5  4  3  5


ANTAL VARIABELEN= 1
WEEBLOKPARTITIE    MSPARTITIE           MIPARTITIE          BLOKNR S    BLOKNR I

ANTAL VARIABELEN= 2
WEEBLOKPARTITIE    MSPARTITIE           MIPARTITIE          BLOKNR S    BLOKNR I
     1 2 1 1 1         1 1 2 1 2          1 1 1 2           2           2

ANTAL VARIABELEN= 3
WEEBLOKPARTITIE    MSPARTITIE           MIPARTITIE          BLOKNR S    BLOKNR I
     1 1 1 2 1         1 1 2 1 2          1 2 3 1           2           3
     1 2 1 2 1         1 1 2 1 2          1 2 3 3           2           3

ANTAL VARIABELEN= 4
WEEBLOKPARTITIE    MSPARTITIE           MIPARTITIE          BLOKNR S    BLOKNR I
     1 1 2 1 1         1 2 3 2 4          1 2 3 2           4           3
     1 1 2 1 2         1 2 3 1 3          1 2 3 3           3           3
     1 1 2 2 1         1 2 3 2 4          1 2 3 4           4           4
     1 1 2 2 2         1 2 3 1 3          1 2 3 4           3           4
     1 2 1 1 2         1 2 1 3 3          1 2 3 4           3           4
     1 2 2 1 1         1 2 3 2 4          1 2 3 4           4           4
     1 2 2 1 2         1 2 3 1 3          1 2 3 4           3           4
     1 2 2 2 1         1 2 3 2 4          1 2 3 4           4           4
     1 2 2 2 2         1 2 3 1 3          1 2 3 3           3           3

ANTAL VARIABELEN= 5
WEEBLOKPARTITIE    MSPARTITIE           MIPARTITIE          BLOKNR S    BLOKNR I
     1 1 1 1 2         1 2 3 4 5          1 2 3 4           5           4
     1 1 1 2 2         1 2 3 4 5          1 2 3 4           5           4
     1 2 1 2 2         1 2 3 4 5          1 2 3 4           5           4
```

J PL/I 018 12•35•16,DURATION 00•07•48

4• Berekening van het PROFIEL•

Dit programma wordt gebruikt in hoofdstuk 5) als procedure en is hier apart weergegeven om een indruk te krijgen van het verloop van de berekende profielen• Na de berekening van een profiel wordt weer het meest gunstige volgende profiel bepaald•
Het verloop van het profiel is voor verschillende configuraties uitgerekend; het maximale aantal variabelen is 4• Het aantal geheugenelementen is resp 2,3,4•

61

```
JOB PL/I,ECB,JANSENPG
OPTION LINK,LIST,SYM
EXEC PL/I
```

09/10/70 11-35-03 036 12•39•59

PROFI GOED.

O P T I O N S L I S T

PROCESS STMT

PTIONS TAKEN ARE 60C,LIST,SYM,ERRS,OPT,STMT.

```
TEST: PROCEDURE OPTIONS(MAIN);

  1              TEST: PROCEDURE OPTIONS(MAIN);
  2              DECLARE (I,J,STATEVAR, PROF(4),A,C,K,TOTVAR,OUDPROF(4))
                           FIXED BINARY(8);
  3              PROFI: PROCEDURE;
  4              DECLARE (LAAG,TOT) FIXED BINARY(8);
  5              A=1;
  6              DO I=STATEVAR TO 2 BY -1;
  7                      IF PROF(I)¬=TOTVAR THEN DO;
  8              GGG:
                         DO J=I-1 TO 2 BY -1;
  9                      IF PROF(J)>PROF(J-1) THEN GOTO FFF;
 10                      END;
 11                      IF PROF(1)=0 THEN GOTO BBB;
 12                      PROF=OUDPROF; A=C; C=C+1; J=1;
 16              FFF:
                                   DO K=STATEVAR-1 TO J+1 BY -1;
 17                                IF PROF(K)<PROF(K+1) THEN DO;
 18                                          PROF(K)=PROF(K)+1; PROF(J)=PROF(J)-1;
 20                                          IF A=1 THEN GOTO EEE;
 21                                          ELSE DO ; A=A-1; GOTO FFF; END;
 25                       END;     END;
 27                                IF PROF(STATEVAR)=TOTVAR THEN GOTO BBB;
 28                                ELSE DO;
 29                                PROF(STATEVAR)=PROF(STATEVAR)+1; PROF(J)=PROF(J)-1;
 31                                IF A=1 THEN GOTO EEE;
 32                                ELSE DO; A=A-1; GOTO FFF; END;
 36              END;    END;      END;
 39              BBB:
                 TOT=SUM(PROF)+1;
 40              LAAG=FLOOR(TOT/STATEVAR);
 41              I=MOD(TOT,STATEVAR);
 42              DO J=1 TO STATEVAR-I;
 43              PROF(J)=LAAG;
 44              END;
 45              LAAG=LAAG+1;
 46              DO J=STATEVAR-I+1 TO STATEVAR;
 47              PROF(J)=LAAG;
 48              END;
 49              OUDPROF = PROF; C=1;
 51              EEE:
                 DO I = 1 TO STATEVAR;
 52              PUT EDIT (PROF(I)) (F(1));
 53              END;
 54              PUT EDIT (' ') (A(2));
 55              END PROFI;
 56              TOTVAR=4;
 57              DO STATEVAR = 2 TO 4;
 58              PROF=0;
 59              PUT SKIP;
 60              AAA: CALL PROFI;
 61              IF PROF(1)¬=TOTVAR THEN GOTO AAA;
 62              END;
 63              END TEST;
```

// EXEC

01 11 02 12 03 22 13 04 23 14 33 24 34 44

001 011 002 111 012 003 112 022 013 004 122 113 023 014 222 123 114 033 024 223 133 124 034 233

224 134 044 333 234 144 334 244 344 444

0001 0011 0002 0111 0012 0003 1111 0112 0022 0013 0004 1112 0122 0113 0023 0014 1122 1113 0222 0123

0114 0024 1222 1123 1114 0223 0133 0124 0034 2222 1223 1133 1124 0233 0224 0134 0044 2223 1233 1224

1134 0333 0234 0144 2233 2224 1333 1234 1144 0334 0244 2333 2234 1334 1244 0344 3333 2334 2244 1344

0444 3334 2344 1444 3344 2444 3444 4444

5• <u>Toekenningen voor een sequentiele machine</u>•

Het nu volgende programma doet toekenningen met een bijbehorende optimale verminderde afhankelijkheid•
Het maximale aantal te verwerken inputs en toestanden (inclusief de gespecificeerde don't cares) is 15•
Meer dan 880 mogelijke toekenningen kunnen niet worden verwerkt•

Een volgorde tabel wordt ingelezen als beschreven onder hoofdstuk 2•

68

```
// JOB PL/I,ECB,JANSENPG                         29/10/70  9-42-44 008  10•30•17
// OPTION LINK,LIST,SYM
// EXEC PL/I
```

69

OPTIONS LIST

```
* PROCESS STMT

OPTIONS TAKEN ARE 60C,LIST,SYM,ERRS,OPT,STMT.
```

```
    1                TWOBLO: PROCEDURE OPTIONS(MAIN);
    2                DECLARE (I,J,Z,C,SNR,INDX(9),D,E,F,TOEWYS(4)) FIXED BINARY(16),
                             (ML,MR,N,TABLE(16,16),TEL,A,STATEVAR,INPUTVAR,TOTVAR,EEN,
                              BNR,CNR,PROF(4),O,OUDPROF(4),L,H,P,K,G,AFPROF(4))
                              FIXED BINARY(4),
                             (GEHAD(16),V,W) BIT(1),
                              (ROWN(16),COLN(16),MSVAR(512),MTVAR(512),ONTHOU,PRODU(16),
                              AVAR,HPRODU(16)) BIT(4),
                             (RY(16,16),KO(16,16),HKO(16),HRY(16),REMAIN,TWOBLOK(512))
                                                                                  BIT(16),
                             (ROW(512,16),HULP8(16)) BIT(8),
                              RYPT(16) BIT(16) BASED(RP), RP POINTER,
                              KOPT(16) BIT(16) BASED(KP), KP POINTER,
                              ROWPT1(16) BIT(8) BASED(RPT1),RPT1 POINTER,
                              ROWPT2(16) BIT(8) BASED(RPT2),RPT2 POINTER;

    3                PROFI: PROCEDURE;

                     /* LEVERT HET VOLGENDE MEEST GUNSTIGE PROFIEL */

    4                DECLARE (LAAG,TOT) FIXED BINARY(8);
    5                A=1;
    6                DO I=STATEVAR TO 2 BY -1;
    7                        IF PROF(I)¬=TOTVAR THEN DO;
    8                GGG:
                             DO J=I-1 TO 2 BY -1;
    9                        IF PROF(J)>PROF(J-1) THEN GOTO FFF;
   10                        END;
   11                        IF PROF(1)=0 THEN GOTO BBB;
   12                        PROF=OUDPROF; A=C; C=C+1; J=1;
   16                FFF:
                                     DO K=STATEVAR-1 TO J+1 BY -1;
   17                                IF PROF(K)<PROF(K+1) THEN DO;
   18                                          PROF(K)=PROF(K)+1; PROF(J)=PROF(J)-1;
   20                                          IF A=1 THEN GOTO EEE;
   21                                          ELSE DO ; A=A-1; GOTO FFF; END;
   25                         END;           END;
   27                                IF PROF(STATEVAR)=TOTVAR THEN GOTO BBB;
   28                                ELSE DO;
   29                                PROF(STATEVAR)=PROF(STATEVAR)+1; PROF(J)=PROF(J)-1;
   31                                IF A=1 THEN GOTO EEE;
   32                                ELSE DO; A=A-1; GOTO FFF; END;
   36                END;    END;        END;
   39                EEE:
                     TOT=SUM(PROF)+1;
   40                LAAG=FLOOR(TOT/STATEVAR);
   41                I=MOD(TOT,STATEVAR);
   42                DO J=1 TO STATEVAR-I;
   43                PROF(J)=LAAG;
   44                END;
   45                LAAG=LAAG+1;
   46                DO J=STATEVAR-I+1 TO STATEVAR;
   47                PROF(J)=LAAG;
   48                END;
   49                OUDPROF = PROF; C=1;
```

```
TWOBLO: PROCEDURE OPTIONS(MAIN);

   51                 EEE:
                      END PROFI;

   52                 CHEK: PROCEDURE;

                      /* CONTROLEER OF DOORSNEDEN VAN PRODUCT(TAU(I))<=M(S-S)(TAU(J)) */

   53                  ONTHOU=HPRODU(1); AVAR=SUBSTR(ROWPT1(1),1,4); GEHAD='0'B;
   56                 GEHAD(1)='1'B; A=2;
   58                 PPP:
                      DO G=A TO ML;
   59                         IF HPRODU(G)=ONTHOU THEN DO;
   60                         GEHAD(G)='1'B;
   61                         IF SUBSTR(ROWPT1(G),1,4)¬=AVAR THEN DO;
   62                                   V='0'B; GOTO QQQ; END;
   65                          END;
   66                 END;
   67                 DO G=A TO ML;
   68                 IF GEHAD(G)='0'B THEN DO;
   69                         A=G;  ONTHOU=HPRODU(G); AVAR=SUBSTR(ROWPT1(G),1,4); GOTO PPP;
   73                         END;
   74                 END;
   75                 V='1'B;
   76                 QQQ: END CHEK;

   77                 GET EDIT (ML,MR,N)(F(2));
   78                 DO I=1 TO ML; DO J=1 TO N; GET EDIT (TABLE(I,J))(F(2));
   81                 PUT EDIT (TABLE(I,J))(X(2),F(2)); END; PUT SKIP; END;

                      /* BEREKEN BASIS RY- EN KOLOMNUMMERS */

   85                 TEL=0;
   86                 DO A=MR TO 2 BY -1;
   87                 TEL=TEL+1;
   88                 DO I=1 TO ML;
   89                 DO J=1 TO N;
   90                 IF TABLE(I,J)=A THEN SUBSTR(RY(TEL,I),J,1)='1'B;
   91                 END;
   92                 END;
   93                 DO I=1 TO N;
   94                 DO J=1 TO ML;
   95                 IF TABLE(J,I)=A THEN SUBSTR(KO(TEL,I),J,1)='1'B;
   96                 END;
   97                 END;
   98                 END;

                      /* BEPAAL WELKE TOEKENNINGEN GEBRUIKT KUNNEN WORDEN */

   99                 STATEVAR=CEIL(LOG2(ML)); INPUTVAR=CEIL(LOG2(N));
  101                 TOTVAR=STATEVAR+INPUTVAR;
  102                 K=2**(STATEVAR-1); C=0; L=16-MR+ML; Z=2**(MR-1)-1;
  106                 DO I=1 TO Z;
  107                 HKO='0'B; HRY='0'B; REMAIN=I; EEN=0; TEL=0; GEHAD='0'B;
  113                 DO J=16 TO L+1 BY -1;
  114                 TEL=TEL+1;
```

72

```
TWOBLO: PROCEDURE OPTIONS(MAIN);

  115            IF SUBSTR(REMAIN,J,1)='1'B THEN GEHAD(TEL)='1'B;
  116            END;
  117            DO J=L TO 18-MR BY -1;
  118            TEL=TEL+1;
  119            IF SUBSTR(REMAIN,J,1)='1'B THEN DO;
  120                    GEHAD(TEL)='1'B; EEN=EEN+1;
  122            END;
  123            END;
  124            IF (EEN>K)|((ML-EEN)>K) THEN GOTO CCC;

                 /* BEREKEN RY- EN KOLOMNUMMERS */

  125            C=C+1; TWOBLOK(C)=REMAIN;
  127            DO J=1 TO MR-1;
  128            IF GEHAD(J)='1'B THEN DO;
  129                    RP=ADDR(RY(J,1)); KP=ADDR(KO(J,1));
  131                    HRY=(HRY|RYPT); HKO=(HKO|KOPT);
  133            END;
  134            END;
  135            A=2; GEHAD='0'B; GEHAD(1)='1'B; RNR=1; ROWN(1)=RNR; REMAIN=HRY(1);
  141            HHH:
                 DO J=A TO ML;
  142            IF HRY(J)=REMAIN THEN DO;
  143                    GEHAD(J)='1'B; ROWN(J)=RNR;
  145            END;
  146            END;
  147            DO J=A TO ML;
  148            IF GEHAD(J)='0'B THEN DO;
  149                    RNR=RNR+1; REMAIN=HRY(J); A=J+1; ROWN(J)=RNR;GOTO HHH;
  154            END;
  155            END;
  156            A=2; GEHAD='0'B; GEHAD(1)='1'B; CNR=1; COLN(1)=CNR; REMAIN=HKO(1);
  162            III:
                 DO J=A TO N;
  163            IF HKO(J)=REMAIN THEN DO;
  164                    GEHAD(J)='1'B; COLN(J)=CNR;
  166            END;
  167            END;
  168            DO J=A TO N;
  169            IF GEHAD(J)='0'B THEN DO;
  170                    CNR=CNR+1; REMAIN=HKO(J); A=J+1; COLN(J)=CNR; GOTO III;
  175            END;
  176            END;
  177            RPT1=ADDR(ROW(C,1));
  178            DO J=1 TO ML;
  179            SUBSTR(ROWPT1(J),1,4)=ROWN(J);
  180            END;
  181            DO J=1 TO N;
  182            SUBSTR(ROWPT1(J),5,4)=COLN(J);
  183            END;
  184            RNR=CEIL(LOG2(RNR)); CNR=CEIL(LOG2(CNR));
  186            MSVAR(C)=RNR; RNR=RNR+CNR; MTVAR(C)=RNR;
  189            CCC: END;

                 /* RANGSCHIK RY- EN KOLOMNUMMERS NAAR AANTAL VARIABELEN */
```

```
 190              Z=1; SNR=1;
 192              DO A=0 TO TOTVAR;
 193              AVAR=A;
 194                      DO I=Z TO C;
 195                      IF AVAR¬=MTVAR(I) THEN GOTO AAA;
 196                      IF I=SNR THEN DO; SNR=SNR+1; GOTO AAA; END;
 200                      RPT1=ADDR(ROW(I,1)); RPT2=ADDR(ROW(SNR,1));HULP8=ROWPT2;
 203                      ROWPT2=ROWPT1; ROWPT1=HULP8;
 205                      REMAIN=TWOBLOK(SNR); TWOBLOK(SNR)=TWOBLOK(I);TWOBLOK(I)=REMAIN;
 208                      ONTHOU=MSVAR(SNR); MSVAR(SNR)=MSVAR(I); MSVAR(I)=ONTHOU;
 211                      ONTHOU=MTVAR(SNR); MTVAR(SNR)=MTVAR(I); MTVAR(I)=ONTHOU;
 214                      SNR=SNR+1;
 215              AAA: END;
 216                      Z=SNR; INDX(A+1)=Z;
 218              END;

                  /* BEPAAL BEGINPROFIEL */

 219              D=1;
 220              DO I=1 TO TOTVAR+1;
 221              IF INDX(I)>STATEVAR THEN DO;
 222                      DO J=D TO STATEVAR;
 223                      PROF(J)=I-1;
 224                      END;
 225                      GOTO DDD;
 226                                END;
 227              ELSE DO;
 228              C=INDX(I)-1;
 229              DO J=D TO C;
 230              PROF(J)=I-1;
 231              END;
 232              D=C+1;
 233              END;
 234              END;
 235              DDD:
 236              E=SUM(PROF); D=FLOOR(E/STATEVAR); I=MOD(E,STATEVAR);
 238              IF D>1 THEN C=D-1; ELSE C=1;
 240              DO J=1 TO STATEVAR-I; OUDPROF(J)=D; END;
 243              D=D+1;
 244              DO J=STATEVAR-I+1 TO STATEVAR; OUDPROF(J)=D; END;

                  /* ZOEK TAU(I) ZDD PRODUCT(TAU(I))=0 */

 247              PRODU='0'B; L=16-MR+ML; I=1;
 250              NNN:
                  E=1;
 251              TTT:
                  F=INDX(PROF(I)+1)-1;
 252              DO J=E TO F;
 253                      DO H=18-MR TO L;
 254                      SUBSTR(PRODU(H),I,1)=SUBSTR(TWOBLOK(J),H,1);
 255                      END;
 256              IF I¬=1 THEN DO;
 257                      P=2**(STATEVAR-I);K=18-MR; GEHAD='0'B; RNR=1; ONTHOU='0'B;
```

```
TWOBLO: PROCEDURE OPTICNS(MAIN);

  262              JJJ:
                             DO H=K TO L;
  263                       IF PRODU(H)=ONTHOU THEN DO;
  264                                 GEHAD(H)='1'B; RNR=RNR+1; IF RNR>P THEN GOTO LLL;
  267                                               END;
  268                       END;
  269                       DO H=K TO L;
  270                       IF GEHAD(H)='0'B THEN DO;
  271                                 ONTHOU=PRODU(H); K=H+1; RNR=1; GOTO JJJ; END;
  276                       END;
  277              END;
  278                       TOEWYS(I)=J;
  279              IF I=STATEVAR THEN GOTO COO;
  280              I=I+1; E=J+1; GOTO TTT;
  283              LLL: END;
  284              G=I; K=2;
  286              UUU:
                   IF I>1 THEN DO;
  287              DO H=18-MR TO L;
  288              SUBSTR(PRODU(H),I,1)='0'B;
  289              END;
  290                       I=I-1;
  291                       IF PROF(I)=PROF(I+1) THEN DO;
  292                                 IF TOEWYS(I)>=INDX(PROF(I)+1)-K THEN DO;
  293                                 K=K+1; GOTO UUU; END;
  296                                 ELSE DO; E=TOEWYS(I)+1; GOTO TTT; END;
  300                       END;
  301                       ELSE DO;
  302                                 IF TOEWYS(I)=INDX(PROF(I)+1)-1 THEN DO;
  303                                 K=2; GOTO UUU; END;
  306                                 ELSE DO; E=TOEWYS(I)+1; GOTO TTT; END;
  310              END;     END;
  312              DO H=1 TO G; APPROF(H)=PROF(H); END;
  315              MMM:
                   CALL PROFI;
  316              IF PROF(1)=TOTVAR THEN DO;
  317              PUT SKIP EDIT ('HET PROGRAMMA IS NOG FOUT') (A); GOTO ZZZ; END;
  320              DC H=1 TC G;
  321              IF PROF(H)¬=APPROF(H) THEN DO;I=1; GOTO NNN; END;
  325              END;
  326              GOTO MMM;

                   /* CONTROLEER OF AFHANKELIJKHEDEN IN OVEREENSTEMMING ZIJN MET HET
                   PROFIEL */

  327              OCC:
                   DO C=1 TO STATEVAR;
  328              J=TOEWYS(O); RPT1=ADDR(ROW(J,1));
  330              RNR=MSVAR(J); CNR=MTVAR(J); P=CNR-RNR; CNR=PROF(STATEVAR)-CNR;
  334              CNR=CNR+RNR;
  335              DC RNR=RNR TO CNR;
  336              IF RNR=STATEVAR THEN GOTO SSS;
  337              IF RNR=0 THEN GOTO SSS;
  338              IF RNR¬=2 THEN DO;
  339                       IF RNR=1 THEN DO; ONTHOU='0'B; W='1'B; END;
```

```
TWOBLO: PROCEDURE OPTICNS(MAIN);

 343                       ELSE DO; ONTHOU='1'B; W='0'B; END;
 347                       DO K=1 TO STATEVAR;
 348                       SUBSTR(ONTHCU,K,1)=W; G=0;
 350                       DO H=17-MR TO L;
 351                       G=G+1; HPRODU(G)=(PRODU(H)&ONTHOU);
 353                       END;
 354                       CALL CHEK;
 355                       IF V='1'B THEN GOTO SSS;
 356                       CNTHOU=¬W;
 357                       END;
 358               END;
 359               ELSE DO;
 360                       CNTHOU='0'B;
 361                       DO K=1 TO STATEVAR-1;
 362                       DO J=K+1 TO STATEVAR;
 363                       SUBSTR(ONTHOU,K,1) ='1'B; SUBSTR(ONTHOU,J,1) ='1'B; G=0;
 366                       DC H=17-MR TO L;
 367                       G=G+1; HPRODU(G)=(PRODU(H)&ONTHOU);
 369                       END;
 370                       CALL CHEK;
 371                       IF V='1'B THEN GOTO SSS;
 372                       ONTHOU='0'B;
 373                       END; END;
 375               END;
 376               END;
 377               E=TOEWYS(I)+1; GOTO TTT;
 379               SSS: AFPROF(O)=RNR+P;
 380               END;
 381               A=1;
 382               VVV:
                   DC C=A TC STATEVAR;
 383                       IF PROF(A)=AFPROF(A) THEN DO;
 384                                IF A=O THEN DO;
 385                                P=AFPROF(O); AFPROF(O)=AFPROF(A); AFPROF(A)=P;
 388                                END;
 389                       A=A+1; GOTO VVV;
 391                       END;
 392               END;
 393               IF A¬=STATEVAR+1 THEN DO;
 394               E=TOEWYS(I)+1; GOTO TTT;
 396               END;
 397               PUT SKIP;
 398               PUT SKIP EDIT ('HET OPTIMALE PROFIEL VOOR DEZE MACHINE IS ',(PROF(I)DO
                   I=1 TO STATEVAR))(A,4(F(1)));
 399               PUT SKIP;
 400               PUT SKIP EDIT ('DE TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PART
                   ITIES ZIJN DAN:')(A);
 401               PUT SKIP;
 402               DC I=1 TC STATEVAR;
 403               J=TOEWYS(I); Z=TWOBLOK(J); RPT1=ADDR(ROW(J,1));
 406               PUT SKIP EDIT (Z,' ')(F(4),A(5));
 407               DO G=1 TO ML;
 408               ONTHCU=SUBSTR(ROWPT1(G),1,4); RNR=ONTHOU;  PUT EDIT (RNR)(X(2),F(2));
 411               END;
 412               PUT EDIT (' ')(A(5));
```

```
TWOBLO: PROCEDURE OPTIONS(MAIN);

  413             DO G=1 TO N;
  414             ONTHOU=SUBSTR(ROWPT1(G),5,4);RNR=ONTHOU; PUT EDIT (RNR)(X(2),F(2));
  417             END; END;
  419             ZZZ:
                  END TWOBLO;
```

// EXEC

77

1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	7	6	3	2	5	4	2
4	2	6	4	2	2	4	2
5	7	2	5	2	5	2	2
1	6	1	1	6	4	1	6
1	1	7	1	7	1	5	7

HET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 336

DE TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

21	1	1	2	1	2	1	3	1	1	2	1	2	1	2	2
26	1	1	2	2	1	3	1	1	2	1	1	2	2	1	2
35	1	2	3	4	5	6	7	1	2	3	1	4	5	6	4

EOJ PL/I 008 10•43•12,DURATION 00•12•54

// EXEC

2	3
4	5
5	6
1	1
1	1
1	1

HET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 233

DE TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

7	1	2	2	1	1	1	1	1
10	1	1	2	3	3	3	1	2
30	1	1	2	3	3	3	1	2

// EXEC

```
  4    1    3
  5    2    4
  1    3    4
  2    3    6

HET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 34

DE TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

 15          1    1    2    2          1    2    3
 20          1    2    3    1          1    2    3
```

// EXEC

3	1	4	2
1	5	4	2
3	4	3	5
5	1	4	2
5	4	3	5

HET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 244

DE TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

8	1	1	2	1	2	1	1	1	2
5	1	2	3	1	3	1	2	3	3
14	1	2	3	2	4	1	2	3	4

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	2	4	6	8	10
3	6	9	2	5	8	1	4	7	10
4	8	2	6	10	4	8	2	6	10
5	10	5	10	5	10	5	10	5	10
6	2	8	4	10	6	2	8	4	10
7	4	1	8	5	2	9	6	3	10
8	6	4	2	10	8	6	4	2	10
9	8	7	6	5	4	3	2	1	10
10	10	10	10	10	10	10	10	10	10

ET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 2247

E TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

33	1	1	1	1	2	1	1	1	1	2	1	1	1	1	2	1	1	1	1	2
341	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
396	1	2	2	1	3	1	2	2	1	3	1	2	2	1	3	1	2	2	1	3
198	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5

```
  2
  3
  1

ET OPTIMALE PROFIEL VOOR DEZE MACHINE IS 12

E TOEKENNINGEN MET BIJBEHORENDE M(S-S)- EN M(I-S) PARTITIES ZIJN DAN:

  1          1   2   1          1
  2          1   2   2          1
```