INFERENCE : Een programma voor grammatische inferentie.

door C. Witteveen.

1. Inleiding

Niet zelden zal een sociaalwetenschappelijk onderzoeker zich voor het probleem geplaatst zien een non-numeriek model te construeren ter verklaring/beschrijving van een reeks onderzochte verschijnselen. In dit kader is de laatste jaren een toenemende belangstelling voor het gebruik van formele talen ter beschrijving van datastructuren op te merken. Grammatica's, bestaande uit een verzameling productieregels, of andere formeel equivalente systemen, worden gebruikt om de structurele relaties in data -gepresenteerd in de vorm van symbolenreeksen- weer te geven. Een grammatica kan echter behalve als een structurele beschrijving van data ook gebruikt worden om een mogelijk oneindige klasse van data te genereren.

Een interessant probleem is, na te gaan of met behulp van een beperkt aantal procedures die van te voren opgesteld zijn, het mogelijk is grammatica's direct te infereren uit de data.

In dit artikel zal een programma besproken worden, dat een grammatica genereert voor een mogelijk oneindige klasse (een taal) van reeksen symbolen als een steekproef van reeksen (zinnen) aangeboden wordt.

Alvorens over te gaan tot beschrijving van het programma zal enige aandacht geschonken worden aan een aantal gangbare begrippen en onderscheidende aspecten die bij de behandeling van inferentieprocessen een rol spelen.

2. Inferentieprocessen

Als inferentieproces kan worden beschouwd ieder proces door middel waarvan algemene conclusies getrokken worden op grond van particuliere gegevens. In dit kader wordt dan ook wel gesproken van een inductief inferentieproces.

Algemeen worden in een (inductief) inferentieproces twee componenten onderscheiden (Kugel, 1977):

- een genererende component, die gerepresenteerd kan worden door een verzameling van procedures die conclusies/theorieën genereren op basis van gerepresenteerde data. Deze component wordt de (inductieve) generator genoemd.

- een evaluatieve component, die gerepresenteerd kan worden door een verzameling procedures die een beoordeling geven van de geïnfereerde conclusies/theorieën in relatie tot de gepresenteerde data. (inductieve logica of evaluator)

Inferentieprocessen kunnen verder onderscheiden worden - met behulp van bovengenoemde aspecten - naar:

- de aard en de presentatiewijze van data
- informatie waarvan de generator, naast de gepresenteerde data, gebruik maakt bij de constructie van theorieën
- de rol die de evaluator speelt in de voortgang van het theorieconstructieproces.

Als voorbeeld bespreken we een aantal mogelijke functies van de evaluator in het inferentieproces.
Als de evaluator geen onderdeel uitmaakt van het genererende inferentiesysteem per se, kan de evaluator de rol vervullen van instructeur in het inferentieproces, zoals bv. in een interactieve leersituatie.
De evaluator bepaalt de presentatie van de data en beoordeelt de conclusies die gegenereerd worden, zodat bijsturing van het inferentieproces mogelijk is.
Een ander uiterste is de situatie waarin de evaluatieve component door middel van een feedbackloop met de generator verbonden is en een filterfunctie vervult.
Een voorbeeld van deze situatie is het welbekende generator-test paar: een inferentieproces, waarin de generator 'theorieën uit een -meestal van te voren vastgelegde- klasse achtereenvolgens opsomt en de evaluator de gegenereerde theorieën beoordeelt volgens vaste criteria.

3. Grammatische inferentie

Een belangrijk onderdeel van het inferentie-onderzoek wordt gevormd door de studie van grammatische inferentieproblemen. Grammatisch inferentie-onderzoek biedt uitstekende mogelijkheden om de eigenschappen van genererende systemen in het inferentieproces te onderzoeken, aangezien in de theorie der formele talen al uitvoerig onderzoek is verricht naar klassen van grammatica's (genererende systemen) en de eigenschappen van de verzamelingen (talen) welke zij voortbrengen. Aanvankelijk richtte het onderzoek zich vooral op het aantonen van de existentie of non-existentie van algoritmen voor de inferentie van grammatica's voor talen, waaruit een eindige subset van zinnen als data gepresenteerd worden. In deze studies, waarvan de publicaties van Gold (1967), Biermann en Feldman (1972) en Feldman (1973) de voornaamste resultaten vormen, werd tevens de invloed van verschillende typen van datapresentatie op het inferentie-proces nagegaan.

Aangezien in deze studies een serie enumeratieve procedures in de generator werden gebruikt om grammatica's uit een welomschreven klasse van grammatica's op te sommen en op deze wijze de eerste grammatica in de opsomming, die consistent is met de informatie die tot op dat moment aangeboden is, te selecteren, wordt wel gesproken van de enumeratieve benadering van het inferentieprobleem. Deze benadering is te vergelijken met de eerder genoemde generator-test situatie.

Een benadering van het grammatisch inferentieprobleem die vooral wat de mogelijkheden voor modelconstructie van reële inferentieprocessen van belang is, is de zogenaamde constructieve benadering. In tegenstelling tot inferentiesystemen met enumeratieve procedures die niet correcte grammatica's in hun geheel verwerpen om de volgende grammatica in de opsomming te selecteren, zijn inferentiesystemen met constructieve procedures in staat om gedeelten van de reeds bestaande grammatica aan te passen als nieuwe informatie niet consistent is met de reeds eerder gegenereerde grammatica. Deze aanpassing vindt plaats door bestaande regels te modificeren, uit te breiden of door restricties te leggen op het gebruik van regels in de grammatica.

Idealiter wordt op deze wijze in het constructieproces de toenemende informatie over de onbekende taal L zichtbaar. Een inferentiesysteem dat bij uitstek geschikt is om de invloed van verschillende wijzen van datapresentatie op het constructieproces na te gaan is het systeem, dat door Knobe & Knobe (1976) ontwikkeld is.
Knobe & Knobe hebben een constructief inferentiesysteem ontworpen, dat in interactie met een instructeur, die steekproefzinnen uit een contextvrije taal L aanbiedt, een expliciete grammatica voor de taal L tracht te construeren. Na aanbieding van een steekproefzin construeert het systeem een partiële grammatica voor L, die in interactie met de instructeur getest wordt. Als de grammatica 'illegale' zinnen produceert, wordt de grammatica zo lang aangepast, totdat alleen zinnen uit L geproduceerd kunnen worden. Na aanpassing beslist de instructeur of er nog meer informatie over L aangeboden dient te worden. Bij presentatie van een nieuwe steekproefzin, wordt door het systeem gebruik gemaakt van de kennis over de taal L, die in de vorm van de partiële grammatica gebaseerd op voorafgaande informatie aanwezig is.
In hun artikel presenteerden Knobe & Knobe een aantal interessante voorbeelden van een constructieproces, dat mede afhankelijk is van de wijze waarop de instructeur zijn informatie aanbiedt.
Een nadeel van de door hen ontwikkelde methode is, dat de inferentie beperkt blijft tot contextvrije grammatica's. Bovendien heeft het inferentiesysteem niet de mogelijkheid om onderdelen van één enkele steekproefzin, die belangrijke informatie kunnen bevatten over de structurele relaties die in de taal L tussen zinnen bestaan, nader te analyseren.

3 INFERENCE

INFERENCE is een interactief computerprogramma*), dat gebruik maakt van een aantal procedures voor constructie van een grammatica voor een (onbekende) taal L op basis van aanbieding van een (eindig) aantal steekproefzinnen uit L. Het programma is voor een gedeelte ontwikkeld op basis van de reeds genoemde methode ontworpen door Knobe & Knobe (1976). Tevens zijn echter een aantal belangrijke procedures toegevoegd om een grammatisch inferentiesysteem GIS te simuleren waarvan de basisprincipes reeds volledig uitgewerkt zijn in Witteveen (1978).
GIS kan beschouwd worden als een uitbreiding van het inferentiesysteem van Knobe & Knobe. Zo is het m.b.v. GIS mogelijk ook niet-contextvrije grammatica's te infereren, d.m.v. constructie van geprogrammeerde contextvrije grammatica's (Rozenkrantz (1969) en Salomaa (1973)), en beschikt GIS over een aantal procedures, die ook onderdelen van steekproefzinnen kunnen analyseren. Voor een volledige beschrijving van deze procedures wordt verwezen naar Witteveen (1978).

Aan de hand van eenvoudige voorbeelden zullen nu een aantal eigenschappen en mogelijkheden van INFERENCE besproken worden.
Als het programma in de aktieve werkruimte geladen wordt, start het programma met de volgende mededeling:

```
WELCOME IN THE PROGRAM INFERENCE
A COMPLETE DESCRIPTION OF THE PROGRAM AND ITS USE
CAN BE GIVEN IF YOU TYPE IN  1
IF YOU WANT TO START IMMEDIATELY, TYPE  2
□:
      2
```

Vervolgens geeft het programma de gelegenheid invoer in te typen. In het eerste voorbeeld voeren we -in een enigszins gewijzigde vorm- de inductieve definities voor welgevormde uitdrukkingen van de propositielogica in:

*) Het programma is geschreven in APL-SV (A programming Language/Shared Variable) en is geïmplementeerd op een IBM 370/158. Van de 128 K bytes waarover de gebruiker in de actieve werkruimte beschikt, wordt 43 K ingenomen door het programma. Voor het gebruik van INFERENCE is geen kennis van A.P.L. vereist.

```
   ENTER SENTENCE   :  P
MORE INPUT  :  1
   ENTER SENTENCE   :  (P'P)
MORE INPUT  :  1
   ENTER SENTENCE   :  (NP)
MORE INPUT  :  0
```

Het programma geeft een samenvatting van de gegeven input en construeert de volgende grammatica:

$S \rightarrow P$
$S \rightarrow [NS]$
$S \rightarrow [SVS]$
$P \rightarrow P$
$[\rightarrow ($
$V \rightarrow V$
$] \rightarrow)$
$N \rightarrow N$

Vervolgens wordt de grammatica op productie van illegale zinnen gecheckt door random generering van zinnen:

```
THIS GRAMMAR CAN BE TESTED BY RANDOM GENERATION OF
STRINGS. LANGUAGE STRINGS ARE COMPARED WITH THE
INFORMATION ABOUT THE LANGUAGE. IF BY RANDOM
GENERATION A STRING IS CONSTRUCTED THAT IS UNKNOWN
THIS STRING IS PRINTED; TYPE  1  IF THE STRING HAS
TO BE ACCEPTED ELSE TYPE 0
```

Het programma genereert een aantal willekeurig gekozen zinnen die de grammatica kan genereren en de instructeur wordt gevraagd deze zinnen te beoordelen.

```
(PV(PV(PV)))
□:
      1
((PVP)V(PVP))
□:
      1
(N(NP))
□:
      1
ALL TESTSTRINGS ARE ACCEPTED; FOR MORE TESTTRIALS
TYPE 1, ELSE TYPE 0
□:
      0
```

Alle testzinnen worden geaccepteerd. De instructeur vindt het niet nodig nog meer zinnen te laten genereren.
INFERENCE antwoordt nu met:

```
THIS RUN IS COMPLETED.
TO GIVE MORE INFORMATION IN THE INFERENCEPROCESS TYPE 1
IF YOU WANT TO STOP ..... TYPE 2
FOR ANOTHER RUN OF THE PROGRAM ..... TYPE 3
□:
      1
```

Van de instructeur hoeft niet verlangd te worden dat hij een expliciete kennis bezitt.a.v. de regelstructuur van de taal die hij het inferentie-systeem wil leren. In ons voorbeeld gaan we na, wat er gebeurt als de instructeur meer informatie over de taal aanbiedt, indien de grammatica voor de taal reeds ontwikkeld is.
Nadat de instructeur te kennen gegeven heeft, dat hij aanvullende informatie wil verstrekken, vraagt INFERENCE hem om invoer:

```
ENTER SENTENCE   :  ((NP)V(NP))
MORE INPUT  :  0
YOUR INPUT WAS
          1 ((NP)V(NP))
```

```
THE STRINGS JUST PRESENTED ARE ANALYSED; IF A STRING
IS PRINTED, YOU ARE ASKED TO INDICATE IF IT CAN BE
ACCEPTED OR NOT
```

INFERENCE gaat nu na of de aangeboden zin ((NP)V(NP)) zodanig te reduceren is, dat er geen gelijke, op elkaar volgende, symbolen in de zin voorkomen. Deze reduceeroperatie is toegevoegd aan het programma om INFERENCE de gelegenheid te geven informatie te verkrijgen uit de opbouw van een steekproefzin om op deze wijze een zo eenvoudig mogelijke grammatica te construeren (zie Witteveen, 1978)

```
((NP)V(NP))
□:
      0
(NP)V(NP))
□:
      0
(NP)V(NP)
□:
      0
```

Met behulp van de reeds bestaande grammatica en de toegevoegde informatie wordt een nieuwe grammatica geconstrueerd:

```
THE GRAMMAR CONSTRUCTED IS

S → P
S → [NS]
S → [SVS]
P → P
[ → (
V → V
] → )
N → N
```

Maar :

ADDITIONAL INFORMATION DIDN'T CHANGE THE RESULTING GRAMMAR;SO IF YOU WANT TO GIVE ADDITIONAL INFORMATION, GIVE EXAMPLES OF SENTENCES THAT CAN NOT BE GENERATED BY THE EXISTING GRAMMAR.
FOR MORE INFORMATION, TYPE 1
IF YOU WANT TO STOP, TYPE 2
FOR A NEW RUN, TYPE 3
□:
2

GOOD BYE AND SEE YOU LATER !

Bovenstaande sessie duurde 10 minuten, en kostte minder dan 4 seconden CPU tijd.
Aan het bovenstaande voorbeeld is te zien, dat INFERENCE een zo algemeen mogelijke grammatica probeert te construeren op basis van steekproefinformatie. De kans is dan groot dat de generalisatie te ver doorgevoerd is.
Als voorbeeld noemen we de situatie waarin de instructeur het programma steekproefzinnen uit $L = \{ A^n : n \geq 1 \} \cup \{ B^n : n \geq 1 \}$ wil aanbieden. Hij biedt als eerste steekproef aan $S_1 = \{ A, AA \}$
INFERENCE construeert de grammatica:

$$\underline{S} \rightarrow \underline{A}$$
$$\underline{S} \rightarrow \underline{S}\underline{S}$$
$$\underline{A} \rightarrow A$$

De grammatica genereert $L_1 = \{ A^n : n \geq 1 \}$. De instructeur besluit aanvullende informatie te geven: $S_2 = \{ BB \}$
Aangezien INFERENCE de bestaande grammatica gebruikt bij de constructie van een nieuwe grammatica, wordt de volgende grammatica verkregen:

$\underline{S} \rightarrow \underline{B}$
$\underline{S} \rightarrow \underline{A}$
$\underline{S} \rightarrow \underline{SS}$
$\underline{A} \rightarrow A$
$\underline{B} \rightarrow B$

De grammatica produceert echter teveel: $L_2 = \{x \in \{A, B\}^+\}$, de verzameling van alle niet-lege rijtjes symbolen over {A, B}.
Bij het testen van de grammatica wordt dan ook een illegale zin geproduceerd: AB.
Als de instructeur te kennen heeft gegeven, dat AB een illegale zin is, antwoordt INFERENCE:

> *THE CONSTRUCTION OF A GRAMMAR BASED UPON SAMPLE INFORMATION ABOUT THE LANGUAGE TO BE INFERRED HAS RESULTED IN A GRAMMAR THAT GENERATES STRINGS NOT BELONGING TO THE LANGUAGE.*
> *THEREFORE, A CORRECTION PROCEDURE IS STARTED, CONSTRUCTING A PROGRAMMED CONTEXT FREE GRAMMAR. IN THIS GRAMMAR THE USE OF PRODUCTION RULES IS CONTROLLED BY SO CALLED >>CONTROL-FIELDS<< DETERMINING THE SET OF RULES THAT CAN BE APPLIED FOLLOWING THE APPLICATION OF A CERTAIN RULE.*
> *IF IN THE CONSTRUCTION PROCESS STRINGS ARE PRESENTED, THEN TYPE 1 IF THE STRING BELONGS TO THE LANGUAGE.*
> *IF NOT, THEN TYPE 0.*

Voor een behandeling van geprogrammeerde grammatica's wordt verwezen naar Rozenkrantz (1969) en Salomaa (1973).
De controlevelden van een productieregel bestaan uit een 'succes field' en een 'failure field'. In een succes field, aangeduid met sigma, van een productieregel f_i komen de labels van die productieregels voor, die, nà succesvolle toepassing van f_i, in aanmerking komen voor de volgende afleidingsstap in de productie van een zin. In een failure field, aangeduid met phi, van een productieregel f_i komen de labels van die productieregels voor die, als productieregel f_i niet toegepast kan

worden, in aanmerking komen voor de eerstvolgende afleidingsstap.
Als een contextvrije grammatica gebruikt wordt als basis voor de constructie van een geprogrammeerde grammatica, dan is het mogelijk een recursief opsombare taal te genereren: De klasse van talen voortgebracht door contectvrije geprogrammeerde grammatica's is de klasse van recursief opsombare talen.
De wijze waarop INFERENCE echter geprogrammeerde grammatica's construeert, maakt het mogelijk grammatica's te construeren voor talen uit een deelverzameling van de klasse van contextgevoelige of type 1 talen.
Op deze wijze is INFERENCE in staat een grammatica te construeren niet alleen voor contextvrije maar ook voor een aantal context-gevoelige talen.

Aangezien het aantal geprogrammeerde grammatica's, dat geconstrueerd kan worden uit een contextvrije grammatica exponentieel toeneemt met het aantal nonterminale productieregels in de contextvrije grammatica, tracht INFERENCE de zoekruimte drastisch te reduceren.
In Witteveen (1978) is een procedure beschreven met behulp waarvan verzamelingen van geprogrammeerde grammatica's op éénvoudige wijze uit te schakelen zijn, zonder iedere mogelijke geprogrammeerde grammatica uitvoerig te testen.
Voor toetsing in interactie met de instructeur komt slechts een beperkt aantal geprogrammeerde grammatica's in aanmerking: na ongeveer 15 CPU seconden besteed te hebben aan de selectie van kandidaat grammatica's antwoordt INFERENCE in ons voorbeeld:

8 *CANDIDATE CONTROLSTRUCTURES ARE GENERATED*

(Ter vergelijking: het aantal mogelijke geprogrammeerde grammatica's dat door INFERENCE geconstrueerd kan worden op basis van de gegeven contextvrije grammatica is 2040)

INFERENCE onderzoekt nu of de meest algemene controlestructuur gelegd op de grammatica, consistent is met de tot nu toe verkregen steekproef-informatie.

Na een paar seconden gerekend te hebben wordt het antwoord gegeven:

THE PROGRAMMED GRAMMAR CONSTRUCTED IS:

	GRAMMAR	*SIGMA*	*PHI*
1.	$\underline{S} \rightarrow \alpha\alpha$	1 4	4
2.	$\underline{S} \rightarrow \underline{B}$	5 2	5
3.	$\underline{S} \rightarrow \underline{A}$	5 3	5
4.	$\alpha \rightarrow \underline{S}$	1 4	1 2 3
5.	$\underline{A} \rightarrow A$	5	6
6.	$\underline{B} \rightarrow B$	6	

(De productieregel $\underline{S} \rightarrow \underline{S}\underline{S}$ is opgesplitst in twee niet recursieve regels $\underline{S} \rightarrow \alpha\alpha$ en $\alpha \rightarrow \underline{S}$ om de selectie van geprogrammeerde grammatica's te vergemakkelijken)

Deze geprogrammeerde grammatica genereert de taal

$L = \{ A^n : n \geq 1 \} \cup \{ B^n : n \geq 1 \}$

Merk echter op dat de contextvrije grammatica:

$\underline{S} \rightarrow \alpha$

$\underline{S} \rightarrow \beta$

$\alpha \rightarrow \alpha\alpha$

$\beta \rightarrow \beta\beta$

$\beta \rightarrow B$

$\alpha \rightarrow A$

ook L genereert.

Met behulp van de constructie van geprogrammeerde contextvrije grammatica's is het inferentieproces echter niet beperkt tot de inferentie van contextvrije grammatica's alleen.

Als voorbeeld nemen we een 'beroemde' contextgevoelige taal, waarvoor geen contextvrije grammatica te construeren is:

$L = \{ A^n B^n C^n : n \geq 1 \}$

Als INFERENCE de volgende steekproef van zinnen wordt aangeboden:

S_1 = { AABBCC, ABC, AAABBBCCC }, wordt tenslotte de geprogrammeerde grammatica geconstrueerd:

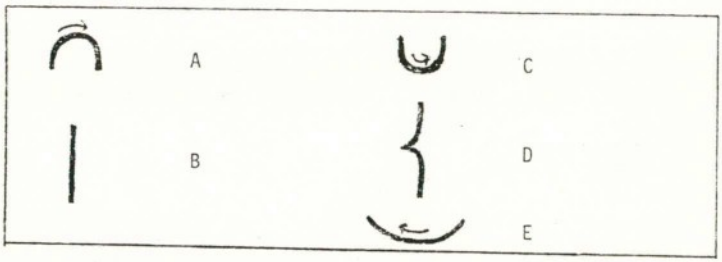
	GRAMMAR	SIGMA	PHI
1.	$\underline{S} \rightarrow \underline{A}\underline{B}\underline{C}$	2 8	ø
2.	$\underline{A} \rightarrow \alpha\alpha$	3	ø
3.	$\alpha \rightarrow \underline{A}$	3	4
4.	$\underline{B} \rightarrow \beta\beta$	5	ø
5.	$\beta \rightarrow \underline{B}$	5	6
6.	$\underline{C} \rightarrow \gamma\gamma$	7	ø
7.	$\gamma \rightarrow \underline{C}$	7	8 2
8.	$\underline{A} \rightarrow A$	8	9
9.	$\underline{B} \rightarrow B$	9	10
10.	$\underline{C} \rightarrow C$	10	ø

4. Inferentie van patronengenererende systemen

Aangezien INFERENCE een aantal subprogramma's bevat die patronen in inputstrings opsporen en delen van inputstrings onderling vergelijken, is nagegaan of INFERENCE geschikt is om patroongrammatica's (die dienen om - mogelijk oneindige - klassen van patronen te beschrijven) te infereren, gegeven een steekproef van patronen uit een welomschreven klasse.

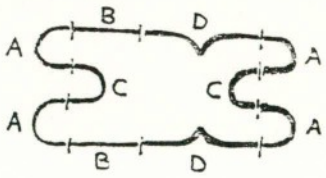
Fu (1974) geeft een aantal contextvrije grammatica's ter beschrijving /generering van klassen van chromosomenpatronen.

Om patronen als een rij symbolen te kunnen coderen, werden patroonsegmenten gecodeerd op de volgende wijze (figuur 1)



Figuur 1. Codering van patroonsegmenten.

Het chromosomenpatroon:



kan nu gecodeerd worden als CABDBACABDBA

Fu onderscheidt in navolging van Ledley et al (1965) de volgende klassen van chromosomen patronen:

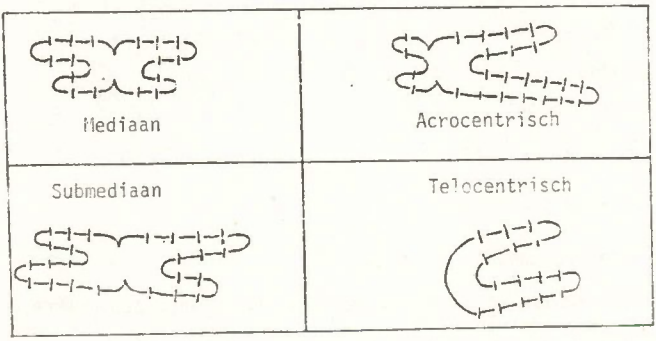
- Mediane chromosomen , code : $CB^{n}AB^{m}DB^{m}AB^{n}CB^{p}AB^{r}DB^{r}AB^{p}$
  $n, m, r, p \geq 0$

- Submediane chromosomen , code : $CB^{n}AB^{m}DB^{p}AB^{r}CB^{s}AB^{t}DB^{v}AB^{w}$
  $n, m, p, r, s, t, v, w \geq 1$

- Acrocentrische chromosomen, code : $CB^{n}AB^{m}DB^{p}AB^{r}CB^{s}AB^{t}DB^{v}AB^{w}$
  met 1. $n, m, p, r, s, t, v, w \geq 0$
  en 2. voor [n, m, p, r] én [s, t, v, w] geldt: de eerste en de tweede óf de derde en de vierde component zijn gelijk aan 0

- Telocentrische chromosomen, code : $EB^{n}AB^{m}CB^{p}AB^{q}$
  $n, m, p, q \geq 0$

In de volgende figuur zijn voorbeelden van chromosomenpatronen uit bovengenoemde klassen weergegeven:



Om een grammatica voor mediane chromosomen te construeren bieden we INFERENCE de volgende steekproef van zinnen aan:

1. CADACADA
2. CBBABBBABBBABBCBABBBDBBAB
3. CBABBBBABBBBBABCBBABBDBBABB

De volgende grammatica G_i wordt gegenereerd:

G_1:

1. $\underline{S} \rightarrow \underline{C}\underline{A}\underline{D}\alpha\epsilon\Delta\rho\omega$
2. $\underline{A} \rightarrow \underline{B}\underline{A}$
3. $\underline{D} \rightarrow \underline{B}\underline{D}$
4. $\alpha \rightarrow \underline{B}\alpha$
5. $\epsilon \rightarrow \underline{B}\epsilon$
6. $\Delta \rightarrow \underline{B}\Delta$
7. $\rho \rightarrow \underline{B}\rho$
8. $\omega \rightarrow \underline{B}\omega$
9. $\omega \rightarrow \omega\underline{B}$
10. $\alpha \rightarrow A$
11. $\epsilon \rightarrow C$
12. $\Delta \rightarrow A$
13. $\rho \rightarrow D$
14. $\omega \rightarrow A$
15. $\underline{A} \rightarrow A$
16. $\underline{C} \rightarrow C$
17. $\underline{D} \rightarrow D$
18. $\underline{B} \rightarrow B$

Deze grammatica genereert niet uitsluitend mediane chromosomen-patronen. Derhalve wordt een geprogrammeerde grammatica gegenereerd, die precies de klasse van mediane chromosomen voortbrengt:

	GRAMMAR	SIGMA	PHI
1.	$\underline{S} - \underline{CAD}\alpha\epsilon\Delta\rho\omega$	2, Π *)	φ
2.	$\underline{A} \rightarrow \underline{BU}$	3	φ
3.	$\underline{U} \rightarrow \underline{A}$	4	φ
4.	$\epsilon \rightarrow \underline{BV}$	5	φ
5.	$\underline{V} \rightarrow \epsilon$	2, 6, Π	φ
6.	$\underline{D} \rightarrow \underline{BW}$	7	φ
7.	$\underline{W} \rightarrow \underline{D}$	8	φ
8.	$\alpha \rightarrow \underline{BX}$	9	φ
9.	$\underline{X} \rightarrow \alpha$	10, 6, Π	φ
10.	$\Delta \rightarrow \underline{BY}$	11	φ
11.	$\underline{Y} \rightarrow \Delta$	12	φ
12.	$\omega \rightarrow \underline{ZB}$	13	φ
13.	$\underline{Z} \rightarrow \omega$	14, 10, Π	φ
14.	$\rho \rightarrow \underline{BT}$	15	φ
15.	$\underline{T} \rightarrow \rho$	16	φ
16.	$\omega \rightarrow \underline{BR}$	17	φ
17.	$\underline{R} \rightarrow \omega$	14, Π	φ

*) Π duidt aan de geordende verzameling terminale regels.

Op dezelfde wijze worden door steekproef'zinnen' uit een klasse van patronen aan te bieden, geprogrammeerde grammatica's voor submediane en acrocentrische chromosomen gegenereerd. Het blijkt nu, dat de geprogrammeerde grammatica's voor mediane, submediane en acrocentrische chromosomen gebaseerd zijn op de grammatica G_1 ; de controlestructuur gelegd op de regels van G_1 verschilt echter per klasse, zoals te zien is in onderstaand schema:

BASISGRAMMATICA	MEDIAAN		SUBMEDIAAN		ACROCENTRISCH	
G_1 :	*SIGMA*	*PHI*	*SIGMA*	*PHI*	*SIGMA*	*PHI*
1. $\underline{S} \rightarrow \underline{CAD}\alpha\epsilon\Delta\rho\omega$	2, Π	ø	2	ø	2, 4	ø
2. $\underline{A} \rightarrow \underline{BU}$	3	ø	3	ø	3	ø
3. $\underline{U} \rightarrow \underline{A}$	4	ø	4, 2	ø	4, 2	ø
4. $\epsilon \rightarrow \underline{BV}$	5	ø	5	ø	5	ø
5. $\underline{V} \rightarrow \epsilon$	2, 6, Π	ø	6, 4	ø	6, 4	ø
6. $\underline{D} \rightarrow \underline{BW}$	7	ø	7	ø	7	ø
7. $\underline{W} \rightarrow \underline{D}$	8	ø	8, 6	ø	2,6,10,14	ø
8. $\alpha \rightarrow \underline{BX}$	9	ø	9	ø	9	ø
9. $\underline{X} \rightarrow \alpha$	10, 6, Π	ø	10, 8	ø	4,8,10,14	ø
10. $\Delta \rightarrow \underline{BY}$	11	ø	11	ø	11	ø
11. $\underline{Y} \rightarrow \Delta$	12	ø	12,10	ø	10,12	ø
12. $\omega \rightarrow \underline{ZB}$	13	ø	13	ø	13	ø
13. $\underline{Z} \rightarrow \omega$	14,10, Π	ø	14,12	ø	12,10, Π	ø
14. $\rho \rightarrow \underline{BT}$	15	ø	15	ø	15	ø
15. $\underline{T} \rightarrow \rho$	16	ø	16,14	ø	14,16	ø
16. $\omega \rightarrow \underline{BR}$	17	ø	17	ø	17	ø
17. $\underline{R} \rightarrow \omega$	14, Π	ø	Π,18,16	ø	14,16, Π	ø

Op deze wijze wordt een link gelegd tussen de drie klassen van patronen. In hoeverre deze beschrijving van patronen adequaat is, is vanzelfsprekend een vraag die met behulp van empirisch onderzoek beantwoord zal moeten worden.

5. Conclusies

In het bovenstaande zijn aan de hand van een aantal voorbeelden enige toepassingen van het programma INFERENCE besproken.
Op dit moment is het niet in de eerste plaats de bedoeling om het programma te verrijken met een aantal extra heuristische procedures om de inferentiecapaciteit op te voeren.
Veel aantrekkelijker lijkt het om een theoretisch fundament te leggen voor heuristische procedures zoals gebruikt in INFERENCE en de eigenschappen van deze procedures te relateren aan eigenschappen van klassen van talen en/of grammatica's.
Als een dergelijke theorie ontwikkeld wordt, kan tevens nagegaan worden, welke perspectieven constructieve inferentie procedures bieden als model voor reële inferentieprocessen zoals probleemoplossen en taalverwerving.