Machinale verificatie van redeneringen

KONINKL. NEDERL. AKADEMIE VAN WETENSCHAPPEN – AMSTERDAM
Overgenomen uit
het Verslag van de gewone vergadering der Afd. Natuurkunde, 78, No. 10, 1969

WISKUNDE

VOORDRACHT

MACHINALE VERIFICATIE VAN REDENERINGEN

DOOR

N. G. DE BRUIJN

Men kan zich als probleem stellen om redeneringen dusdanig nauwkeurig weer te geven dat ze vervolgens geheel automatisch kunnen worden geverifieerd. Daartoe is nodig dat ze in een nauwkeurig omschreven taal worden geschreven, en dat precies wordt aangegeven welk gebruik van deze taal als correcte redenering geldt. In het bijzonder kan men de eis stellen dat een groot (doch nauwkeurig afgebakend) deel der wiskunde op een dergelijke manier kan worden vastgelegd.

Het is niet moeilijk te zeggen wat onder verificatie moet worden verstaan: Er dient een programma te zijn dat een computer in staat stelt om op een in gecodeerde vorm aangeboden tekst met „correct" of „niet correct" te reageren. Daarnaast kan men de praktische eis stellen dat de verwerkingstijd en nodige geheugenruimte binnen redelijke perken blijven wanneer men verder en verder de wiskunde binnendringt.

Men moet in het oog blijven houden dat het raamwerk der geformaliseerde wiskundige redeneringen niet hetzelfde is als „de wiskunde". De machine die de geformaliseerde redeneringen heeft doorgewerkt en beaamd, heeft er nog bitter weinig van begrepen. Hij zal misschien in staat zijn het gelezene te onthouden en later te gebruiken om nieuwe teksten te keuren, maar hij heeft niets begrepen van motiveringen en interpretaties. Men zal hem misschien met enige moeite wat creatief vermogen kunnen geven, maar hij zal daarbij niet geleid worden door ideeën uit de aanschouwingswereld, en evenmin door smaak of gevoel voor waarde. Het geformaliseerde raamwerk is slechts een armzalig deel der wiskunde; niettemin hebben de wiskundigen door de eeuwen heen getracht hun beschouwingen in een dergelijk formalisme, vrij van onzekerheden, te laten kristalliseren. De neergeschreven redenering is de afsluiting van het denkproces, en niet het denkproces zelf.

Ook nog in een ander opzicht kan men zeggen dat volledig geformaliseerde wiskunde een beperking inhoudt. Herhaaldelijk doet het zich voor dat wiskundigen die in een bepaalde taal redeneringen houden, uitspraken gaan doen over de wijze waarop zulke redeneringen in die taal worden uitgedrukt; daarmee komen ze tot resultaten die in de oorspronkelijke taal niet, of minder gemakkelijk konden worden verkregen. Het spreken *over* een taal kan niet in die taal zèlf geschieden, maar in een z.g. *metataal*. Vaak zullen de uitspraken in de metataal erg lijken op uitspraken in de

taal zèlf; dan is bijzondere oplettendheid geboden, want men mag ze niet met elkaar verwarren. Verschillende „paradoxen" kunnen daaraan worden toegeschreven. Als Achilles in het punt P_1 gekomen is waar zoëven de schildpad was, dan is inmiddels de schildpad in P_2; als Achilles in P_2 is gekomen dan is de schildpad in P_3, enz. Er komt geen einde aan. De paradox ontstaat wanneer men de conclusie „er komt geen einde aan" gaat uitleggen als „er komt geen einde aan de achtervolging". Het „er komt geen einde aan" is een meta-uitspraak, en betekent: „er komt geen einde aan ons gesprek over de achtervolging". Wanneer men vermenging van taal en metataal vermijdt (en zich bovendien van cirkeldefinities onthoudt) verdwijnen de meeste „paradoxen" als sneeuw voor de zon.

Toch is het op verstandige manier hanteren van metataal in de wiskunde goed bruikbaar. Niet altijd zal dit een wezenlijke uitbreiding van de te behalen resultaten betekenen; vaak is het slechts een kwestie van efficiëntie. Wanneer men bijvoorbeeld een beschouwing beëindigt met de volgende zinswending: „door het bovenstaande bewijs nog eens te herhalen, maar met verwisseling van de letters p en q, ziet men in dat . . .", dan is zulks efficiënt gebruik van metataal dat in principe vermeden zou kunnen worden.

Een duidelijk voorbeeld van wèl essentiële uitbreiding van de taal is het functiebegrip. Honderden jaren lang is dat een meta-begrip geweest. Men zei dingen als: „laat op de een of andere manier, onverschillig hoe, een uitdrukking of voorschrift gegeven zijn dat . . .". Pas in moderne tijd is het functiebegrip in de taal geïncorporeerd.

Het vastleggen van het wiskundige apparaat met één vaste taal betekent ongetwijfeld een zekere verarming. Het is zaak bij het kiezen van een taal de nadelen van die verarming zo beperkt mogelijk te houden.

Het kiezen van een dergelijke taal wordt bemoeilijkt door het feit dat tot nu toe slechts losse stukken van de wiskunde min of meer geformaliseerd zijn en dat er geen algemeen geaccepteerde opvattingen over complete formalizering bestaan.

In de jaren 1967–1969 werd te Eindhoven de taal Automath ontwikkeld. Dit is een primitieve doch flexibele taal, gebouwd op drie grondpijlers: blokstructuur, toekenning van een type aan elk object, en functionele abstractie. De grammatica van Automath bevat nauwelijks iets dat doet denken aan redeneerpatronen en logica. Wat men van deze zaken nodig heeft kan men schrijven in het begin van het boek waarin men de gehele wiskunde wil schrijven, en men kan het zó presenteren dat het te allen tijde kan worden gebruikt.

De regels waaruit een Automath-boek bestaat, hebben alle deze vorm:

> In de context (1) is (2) een door (3) gegeven ding
> van de soort (4).

Hierin is (2) een nieuwe naam; in elke regel wordt dus één nieuwe naam geïntroduceerd, een z.g. *identificator*. Men kan deze namen vrij kiezen,

mits onderling verschillend. Op de plaats (3) staat òf het symbool „*PN*" (primitive notion), òf het symbool „ – ", òf een expressie, op voorgeschreven wijze opgebouwd met behulp van eerder ingevoerde identificatoren. In het geval van „ – " zeggen we dat de identificator van die regel een „variabele" is. Op de plaats (4) staat òf een expressie òf het symbool „type". Op de plaats (1) staat òf een eerder ingevoerde variabele vermeld, òf een symbool „*O*".

De contextindicatie (1) beoogt te beschrijven welke variabelen bij het lezen van de betreffende regel als toegankelijk worden beschouwd. De contextindicatie geeft weliswaar slechts één variabele aan, maar men kan in het boek terugzoeken wat dáárvan de context is, enz. In het geval van contextindicatie „*O*" zijn geen variabelen beschikbaar. Met voorliefde zal men proberen de regels in overzichtelijke vorm te schrijven zodat voor elke variabele geldt dat de regels waarin hij beschikbaar is een samenhangend stuk tekst (een „blok") vormen, niet onderbroken door regels waar hij niet beschikbaar is. De blokken zijn dan „genest": twee blokken liggen òf geheel buiten elkaar, òf één ervan ligt geheel binnen de andere.

In een gewone wiskundige tekst kan men doorgaans deze blokstructuur herkennen. De blokken worden op twee verschillende manieren geopend. In de eerste plaats kan de variabele een „object" aanduiden. De eerste regel van het blok kan bijvoorbeeld luiden „zij x een reëel getal", of „zij V een verzameling van gehele getallen". Binnen het gehele blok is dan x resp. V beschikbaar. Men kan ook zeggen dat ze in dat stuk tekst als constanten worden behandeld.

In de tweede plaats kan echter een blok geopend worden met een onderstelling. Dit betekent dat binnen het gehele blok die onderstelling als een waarheid wordt behandeld, terwijl buiten dat blok de onderstelling krachteloos is.

In Automath worden deze twee verschillende mogelijkheden op dezelfde manier behandeld. Dikwijls staat bij de regel die het blok opent in de soortkolom, d.i. op plaats (4), niet zoiets als „reëel getal", „punt", „verzameling", maar een uitspraak van het type „$a=b$", „punt P ligt in vlak V", „hoek BAC is recht". Ook in andere regels kunnen zulke uitspraken in de soortkolom staan. Staat er dan op plaats (3) een expressie, dan kan die als „bewijs" van de uitspraak geïnterpreteerd worden; staat er „*PN*" dan is de uitspraak als axioma ingevoerd; staat er „ – " dan is de uitspraak een onderstelling. De identificator is de naam die men moet aanroepen om de uitspraak later te kunnen gebruiken.

Buiten een blok mogen de in dat blok ingevoerde variabelen niet meer worden gebruikt; in het bijzonder zijn de in het blok ingevoerde onderstellingen buiten het blok onbruikbaar. De „gewone" regels uit een blok (d.w.z. de regels die niet een variabele invoeren) zijn buiten het blok wèl bruikbaar, mits men voor de variabelen expressies levert die wèl toegankelijk zijn en bovendien de goede soort hebben. In het bijzonder is alles wat slechts onder zekere onderstelling is afgeleid later weer bruikbaar

als men (eventueel onder gewijzigde omstandigheden) een bewijs voor die onderstelling levert.

Men kan de identificatoren beschouwen als afkortingen (soms afkortingen van bijv. algebraïsche expressies, soms afkortingen van lange argumenten). Zo kan men gaandeweg zeer complexe situaties opbouwen met regels die elk afzonderlijk redelijk eenvoudig zijn.

Het is in het kader van deze voordracht niet mogelijk om de grammatica van Automath in details te bespreken. We volstaan met de opmerking dat deze grotendeels op natuurlijke wijze voortvloeit uit de algemene boven reeds vermelde principes. Wij verwijzen naar [1] en [2].

De computer die een Automath-tekst leest zal nooit in het boek behoeven te zoeken naar de informatie die nodig is om een regel te begrijpen. De regels zijn zó geschreven dat daarin vastligt waar die informatie te vinden is en hoe die moet worden gebruikt.

Hoewel het schrijven van wiskunde in Automath heel goed praktisch uitvoerbaar is, blijft het vooralsnog een moeizame aangelegenheid. Men zou het enigszins kunnen vergelijken met het schrijven van een computerprogramma in de opdrachtcode van de machine, zonder gebruikmaking van enige kunsttaal.

Voor een deel is de moeite die het kost om wiskunde in Automath te schrijven een gevolg van het feit dat men niet gewend is om wiskunde altijd gedetailleerd weer te geven. Een wiskundige auteur zal doorgaans volstaan met een schets van een argumentatie, als hij het vertrouwen heeft dat de lezer de details zèlf kan produceren. Dit dwingt de lezer nl. tot activiteit; zonder die activiteit zal hij het lezen niet lang volhouden. Het is meestal geen luiheid als de auteur uiterste volledigheid vermijdt – integendeel, volledigheid laat zich vaak gemakkelijker schrijven dan leerzame onvolledigheid. De verifiërende computer stelt echter geheel andere eisen aan de volledigheid van het betoog.

Men kan twee wegen aanwijzen om het schrijven in Automath gemakkelijker te maken. In de eerste plaats kan men proberen vaak optredende moeizame wendingen te herkennen en af te korten tot korte opdrachten die een computer in Automath-regels omzet. Een verdergaande mogelijkheid is dat men een computer probeert te leren hiaten in redeneringen zelfstandig te overbruggen. Dit zal geenszins gemakkelijk zijn, maar het zal in de toekomst van grote waarde kunnen worden voor redeneren in samenwerking tussen man en machine.

Een kleine stap in de richting van samenwerking tussen man en machine is aan de T.H. Eindhoven reeds gezet doordat (binnen het T.H.E. multiprogrammeringssysteem) de mogelijkheid bestaat om Automathteksten regel voor regel aan de computer mede te delen, terwijl na elke foutieve regel de computer een uitvoerige diagnose van de gemaakte fout aflevert.

LITERATUUR

1. Bruijn, N. G. de, Automath, a language for mathematics. Rapport 68-WSK-05 (1968) Technische Hogeschool Eindhoven.
2. ———, The mathematical language Automath, its usage, and some of its extensions. Zal verschijnen als onderdeel van: Proceedings of the Symposium on Automatic Demonstration (IRIA, Versailles, December 1968), in de Springer Lecture Notes Series.