2. Syntaxis en semantiek

In dit hoofdstuk worden de begrippen syntaxis en semantiek behandeld. Verder gaan we in op de fouten die hierin gemaakt kunnen worden en waarom dit in de algoritmiek zo desastreus kan zijn. Een exacte definitie is noodzakelijk, omdat de computer menselijke taal niet kan begrijpen.

2.1 Syntaxis en Semantiek

De grammatica van een natuurlijke taal bepaalt hoe de relaties tussen de woorden moeten zijn om correcte zinnen te vormen. Het begrijpen en goed toepassen van deze regels is noodzakelijk om in een bepaalde taal met iemand een gesprek te kunnen voeren. Deze grammatica regels worden ook wel de syntaxis van een taal genoemd.

Een voorbeeld van een syntactisch correcte zin is:

De vogel eet de regenworm op.

Een syntactisch niet correcte zin is bijvoorbeeld:

de eet vogel regenworm de op

Deze zin begint niet met een hoofdletter en eindigt niet met een punt en de schrijver gaat daarmee al twee keer in tegen de afspraken die we met elkaar hebben gemaakt over hoe in de Nederlandse taal een goede zin er uit moet zien. Bovendien staan onderwerp en gezegde niet in de goede volgorde en dat maakt de zin totaal onbegrijpelijk.

De **syntaxis** van een taal is *de verzameling van regels die specificeren hoe zinnen uit letters, cijfers en andere tekens worden samengesteld*. **Natuurlijke** talen, zoals het Nederlands, onderscheiden zich van **formele** talen, zoals wiskunde en programmeertalen, doordat hun verzameling van regels weinig strict zijn.

Zinnen in een natuurlijke taal hebben een **structuur (syntax)** die grotendeels onafhankelijk is van hun **betekenis (semantiek)**. Het belangrijkste argument dat taalwetenschappers (bijv. Chomsky) naar voren brengen om deze bewering te onderbouwen is het feit dat er zinnen bestaan die grammaticaal correct zijn, maar niettemin geen enkele betekenis lijken te hebben:

colorless green ideas sleep furiously

Natuurlijke talen bezitten de eigenschappen van:

- *Dubbelzinnigheid*

In de meeste natuurlijke talen zijn er veel woorden die meerdere betekenissen hebben. Welke van de meerdere betekenissen er precies in het geding is moet dan uit de kontekst blijken.

- *Overbodigheid* (redundantie)

  Om het effect van het bestaan van meerdere betekenissen te kompenseren en om misverstanden te voorkomen gebruiken we vaak meer dan 1 zin om een boodschap over te brengen. Zinnen kunnen op veel manieren geformuleerd worden en toch in grote lijnen dezelfde boodschap bevatten. Aan de andere kant kan het toevoegen of weglaten van ogenschijnlijk onbelangrijke woordjes in een zin een mededeling juist harder of zachter over laten komen. Als iemand een verhaal navertelt dat hij van een ander gehoord heeft, is het hoogst onwaarschijnlijk dat hij exact dezelfde zinnen gebruikt.

- *Letterlijkheid*

  In natuurlijke talen worden spreekwoorden, zegswijzen en beeldspraak gebruikt om bedoelingen duidelijk te maken, zonder dat we dat wat gezegd wordt of geschreven staat, letterlijk moeten nemen.

Formele talen hebben bovenstaande drie eigenschappen pertinent niet, en de regels over welke symbolen geldig zijn, hoe met symbolen omgegaan moet worden en hoe de structuur van zinnen moet zijn, zijn zeer strict. Zo is bijvoorbeeld:

3 + 3 = 6

een foutloze mathematische uitspraak, maar

3 = + 6$

is dat niet ! Het teken $ is geen geldig teken in wiskundige zinnen, en de structuur van de zin (volgorde van de symbolen) klopt niet (een plus teken komt nooit direct na een is-gelijk teken). Om een voorbeeld uit de scheikunde te noemen:

H2O

is een syntactisch foutloze chemische naam, maar

2Zz

is dat niet. De elementen Z en z bestaan niet, de verbinding Zz dus ook niet, en namen van chemische verbindingen beginnen niet met een cijfer.

De **betekenis** van een zin in een taal wordt ook wel de **semantiek** van de taal genoemd. Ook dit is een noodzakelijk onderdeel van de taal. De betekenis van de zin kan pas blijken als we alle woorden van de zin hebben gelezen of gehoord, en de structuur van de zin tot ons is doorgedrongen. Er is een duidelijk verschil tussen de syntaxis en de semantiek. Een syntactisch correcte zin als:

De regenworm eet de vogel op.

is semantisch gezien niet correct. Een zin als:

Geef de naam van de achtste dag van de week.

heeft geen betekenis en is dus semantisch gezien fout (want de achtste dag bestaat niet).

Programmeertalen zijn net als wiskunde **formele talen**. Zo kunnen de regels van de syntaxis van een programmeertaal specificeren dat elk *haakje openen* in een rekenkundige bewerking overeen moet komen met een *haakje sluiten*; of dat elke opdracht moet worden afgesloten met een punt-komma. Semantische regels in programmeertalen bepalen net als in een natuurlijke taal de 'betekenis' van een syntactisch correct programma.

Eén van de manieren om de semantiek van een programmeertaal te beschrijven is het geven van een beschrijving van elke geldige constructie in de taal. Daarmee introduceren we een nadeel doordat de dubbelzinnigheid en wijdlopigheid die elke natuurlijke taal bezit, behouden blijft. Het voordeel is dat er een redelijk intuïtief beeld van de taal ontstaat waarmee men makkelijker omgaat dan met een beschrijving in een streng formele notatie.

Ook in zinnen die syntactisch en semantisch volledig correct zijn kan toch nog een fout zitten. Denk bijvoorbeeld aan:

De oppervlakte van een cirkel is pi maal de straal.

Deze zin is syntactisch en semantisch correct (de woorden hebben een betekenis en staan in de juiste volgorde). De enige fout in deze zin is een **logische** fout (de oppervlakte is namelijk pi maal de straal in het kwadraat).

In de algoritmiek is het zeer belangrijk om een exact gedefinieerde syntaxis en semantiek te gebruiken. De menselijke taal is namelijk veel te ingewikkeld om door een computer begrepen te worden. Programmeertalen zijn exact gedefinieerde, formele talen die een computer begrijpen kan. In de meeste programmeertalen controleert de computer op syntac-

tische en semantische fouten, hoewel niet alle semantische fouten gevonden kunnen worden. De logische fouten zijn nooit door de computer te vinden. De meeste fouten die achteraf nog in programma's blijken te zitten zijn in vrijwel alle gevallen logische fouten.

In het algemeen kan men zeggen:

- Syntactische fouten zijn fouten die de computer meteen vindt.
- Semantische fouten zijn fouten waar het programma tijdens de uitvoer op vast kan lopen. (Let op: het kan soms ook goed gaan)
- Logische fouten zijn fouten waar het programma niet op vastloopt, maar die ervoor zorgen dat de uitvoer niet klopt.