Over stapeladministratie.

1. Binnenblokken, procedureblokken en blokhoogte.

Het begrip blok is louter lexicographisch. Het hele programma is per definitie een blok, binnen in een blok kunnen een of meer andere blokken gedefinieerd worden. Behalve de ALGOL-blokken voeren wij nog in:
a) een procedure-declaratie is per definitie een blok
b) een for-statement is per definitie een blok.

Wij onderscheiden nu twee soorten blokken:
a) procedureblokken
b) alle overige blokken; deze noemen wij "binnenblokken".

Opm. Wij laten voorlopig even buiten beschouwing of we de procedure body als binnenblok van het procedureblok beschouwen.

De aldus ingevoerde blokken hebben de eigenschap, dat elk blok met uitzondering van het programma-blok in een ander blok gedefinieerd is. We voeren nu in het begrip "blokhoogte", dwz. het programma-blok krijgt blokhoogte 0, elk ander blok krijgt een blokhoogte 1 hoger dan die van het lexicographisch onmiddellijk omvattende blok.

2. Binnentophoogte en display.

Aan elk procedureblok kennen we toe een zg. "binnentophoogte". Ook de binnentophoogte is een zuiver lexicographisch begrip: de binnentophoogte is 1 meer dan de maximum blokhoogte in het inwendige van het procedureblok zonder dat het in een daarin X inwendig procedureblok zit. (Andere definitie: het is 1 meer dan de maximum inwendig voorkomende blokhoogte na verwijdering van alle erin voorkomende proceduredeclaraties.)

Binnenblokken hebben de eigenschap, dat men ze slechts vanuit het onmiddellijk omvattende blok kan binnengaan, dit in tegenstelling tot procedureblokken.

Bij binnenkomst van een procedureblok, en alleen dan, zullen we een nieuwe display invoeren. De lengte van deze display kiezen we gelijk aan de binnentophoogte, dwz. groot genoeg om te garanderen, dat we erin de ruimte hebben om (zie onder) de binnenblokken van deze procedure te kunnen behandelen.

De opmerking is nl. dat de techniek van de MC1-vertaler voor ingaan en uitgaan van een binnenblok echt wel efficient is. De MC1-vertaler werkt met een vaste display. Bij binnenkomst hoef je alleen een nieuw element in de display in te vullen en de stapelwijzer op te hogen, bij verlating hoef je alleen de stapelwijzer af te lagen (en displayelement te negeren, maar dat kost niet zo'n moeite.)

Hebben we echter, als in de MC1-vertaler een enkele display, dan hebben we om de haverklap de plicht tot de operatie UDD (Up Date Display). De bewering is, dat slechts een nieuwe display invoeren bij binnenkomst van het procedureblok voldoende is om alle UDD-operaties kwijt te zijn -dwz. te kunnen condenseren tot wisseling van L.

3. De werkruimtewijzer WW.

Tussen statements keert de stapelwijzer SW in een blok steeds terug tot hetzelfde rustpunt; we noemen dit de werkruimtewijzer WW: hij wijst naar de eerste stapelplaats voor de anonyme tussenresultaten.

Normaliter hoeft men zich over de WW-waarde van een blok niet te bekommeren. Tijdens de uitvoering van een statement wordt immers de SW evenveel opgehoogd als afgelaagd en de SW keert automatisch tot zijn rustpunt terug. We hebben echter ook de zg. "plotselinge blokverlating": in een blok kan een"goto L" staat, waarbij "L" een globale label is. Als we hierin terugkeren in een blok, dat via een type procedure, dwz. midden in een statement, tijdelijk verlaten hebben, dan hebben we in het bestemmingsblok als regel nog een paar anonyme tussenresultaten te vergeten, maw: SW moet teruggezet worden op de waarde van WW, die bij de activering van het bestemmingsblok behoort.

Onze gedachten bepalen we even tot binnenblokken. Als we zitten in een binnenblok B van hoogt n, dan zijn we daarin gekomen via het nauwst lexicographisch omvattende procedureblok op hoogte m (m < n). Bij de binnenkomst in dat procedureblok A hebben we in de stapel een nieuwe display ingevoerd, waarvan de onderste m elementen uit een andere display gecopieerd zijn (zie onder),waar het m + 1ste element uit de SW is afgeleid en waarvan de overige elementen beantwoorden aan de binnenblokken van A. Het idee is, om tijdens de uitvoering van binnenblok B de locale grootheden van dit blok te localiseren ten opzichte van element n van deze display en element n + 1 gelijk te laten zijn aan de bijbehorende waarde van WW. Gaan we een binnenblok van B in, dan fungeert element n + 1 onveranderd als referentiepunt voor de locale grootheden van dit blok en element n + 2 bergt dan de (binnen-)WW. Dit is de reden, waarom de binnentophoogte 1 hoger gekozen is dan de maximum voorkomende binnenblokhoogte.

4. Ingaan en uitgaan van binnenblokken.

Bij het ingaan van een binnenblok op hoogte n en bij het uitgaan moeten we de waarde van n weten: bij het binnengaan om te weten waar (nl. op plaats n + 1 van de display) we de nieuwe WW invullen, bij het verlaten omdat we dan de SW terug moeten zetten, dwz. gelijk moeten maken aan element n van de heersende display.

In de programmatekst is de heersende blokhoogte, wat een lexicographisch bepaalde grootheid is, desgewenst statisch bekend en we kunnen de mechanismen "binnenblok in" en "binnenblok uit" creeren, zodat ze van het programma een parameter meekrijgen, dwz. "Binnenblok in van hoogte n" en "binnenblok uit van hoogte n".

Een alternatieve oplossing is, om "ergens" -waarover dadelijk meer- het systeem de heersende blokhoogte bij te laten houden: "Binnenblok in" en "Binnenblok uit" krijgen dan geen parameter van het programma mee, maar raadplegen en wijzigen zelf de "ergens" te vinden heersende blokhoogte. Dit laatste is een techniek, die mij iets meer aanspreekt, hoewel ik er op het moment slechts vage argumenten ten gunste van weet te bedenken. Het is het algemene argument, dat je in je programma niet hoeft te vermelden, wat het verwerkende systeem uit de structuur kan afleiden. (Op precies dezelfde manier als we de waarde van de stapelwijzer niet expliciet aangeven, maar door de stapelende opdrachten laten meesjouwen.) De tekst van het programma wordt door deze technieken korter. Een nevenvoordeel is, dat de inhoud van de stapel, door de heersende blokhoogte daarin onder te bregen, sprekender wordt. (Dit is voorlopig ook nog vaag: het zou wel eens gewenst kunnen zijn, dat we in de stapel een zodanige administratie bijhouden, dat de stapelinhoud op elk gewenst moment "extern interpretabel" is. Ik denk aan stapelverschuivingen door de coordinator en post mortem dumps.)

Als we de heersende blokhoogte willen onthouden, dan rijst de vraag: waar?

In de MC1-vertaler deden we dit op een vaste plaats: daar was de heersende blokhoogte een in enkelvoud aanwezige, plaatsgebonden staturvariabele. Dit zou voor ons onpractisch zijn in verband met multiprogrammering, omdat we dan bij wis-

seling van programma de heersende blokhoogte zouden moeten redden. Als we de heersende blokhoogte als globale variabele van een programma invoeren, dwz. onder in de stapel van dat programma, dan hebben we de multiprogrammeringsmoeilijkheid ondervangen, maar zijn we wat betreft de uni-programmering nog net in de toestand die we bij de MC1-vertaler hebben: redden en herstellen bij procedure aanroepen en returns. Mijn voorlopige suggestie is, om de heersende blokhoogte op te bergen bij de heersende display, dwz. als ML[-1]. Deze suggestie is voorlopig, omdat ik nog moet nakijken, of we door de nog niet in het beeld betrokken parametersituatie niet in moeilijkheden komen.

De heersende blokhoogte kan dan bij ingaan en uitgaan van binnenblokken bijgehouden worden; bij binnenkomst in een procedure, waar we een nieuwe display introduceren, moet de blokhoogte van de procedure bekend zijn (nl. om te weten, hoeveel plaatsen in de nieuwe display gecopieerd moeten worden); bij die gelegenheid kan ook de initialisering van het nieuwe heersende bloknummer verzorgd worden.

Ik eindig deze korte notitie met het signaleren van de volgende mogelijke complicaties:

1) bij blokverlating moeten we, als er in dit blok "grote arrays" geintroduceerd waren, de door deze arrays bezette trommelpagina's weer vrijgegeven worden.

2) bij "plotselinge blokverlating" kunnen een heleboel blokken uit de stapel verdwijnen; wederom geldt, dat aflagen van SW niet voldoende is, omdat er ook grote arrays hierdoor afgevoerd kunnen moeten worden.

3) in de parameter-situatie zitten we nu heel anders: we werken dan onder controle van een heersende L, maar het daarbij geborgen blokhoogte-gegeven ML[-1] heeft op dat ogenblik geen betekenis. Dit zou zich kunnen wreken, als we er over denken, wat we moeten doen, als de stapel verschoven moet worden.

E.W.Dijkstra