Projectwijzer OGO 2.1 (2R660)

Het decoderen van geheimschrift

Vigenère en Enigma

Projectcoördinator: J.F. Groote

Opleiding Technische Informatica 2002–2003

Te doen:
Maken met voorbeeldtekst. Maken Enigmateksten. Regelen unite accounts. Regelen accounts sgtt08. Maken eerste college. Regelen vergadertrainingen. Maken groepsindelingen.

Inhoudsopgave

1 Inleiding

Het doel van dit project[1] is om het geheimschrift van de Enigma te breken. Hiermee wordt ervaring opgedaan in het omgaan met een omvangrijk algoritmisch probleem.

De Enigma is een versleutelmachine die in het begin van de twintigste eeuw werd geconstrueerd. De Enigma heeft vele varianten gekend waarbij de bekendsten in gebruik waren bij het Duitse leger. Hoewel de Duitsers de Enigma voor onkraakbaar hielden waren de Polen er voor de oorlog al in geslaagd de code te breken. Zij hebben vlak voor de Poolse inval de kennis over de Enigma gedeeld met met name de Engelsen. De Engelsen zijn mede hierdoor in staat geweest gedurende vrijwel de hele oorlog de geheime berichten van de Duitsers te decoderen. Het breken van geheimschrift was voor de Engelsen een belangrijke activiteit waaraan aan het eind van de oorlog 10.000 mensen werkten. Daaronder was met name Alan Turing, vader van het begrippen 'berekenbaarheid' en 'Turing test'. Het decoderen van het Duitse geheimschrift is een belangrijke impuls geweest voor de ontwikkeling van de hedendaagse computer. Het belang van Turing voor de informatica is zo groot dat de belangrijkste onderzoeksprijs in de informatica, de Turing Award, naar hem vernoemd is.

Julius Caesar gebruikte al een eenvoudige vorm van geheimschrift, die daarom de Caesar codering wordt genoemd. De basisgedachte hierachter is dat letters uit het alfabet k posities over het alfabet worden opgeschoven. Verschuiving over 0 posities heet een verschuiving over `A`. Verschuiving over 1 positie heet een verschuiving over `B`, etc. Verschuiving over `B` leidt tot:

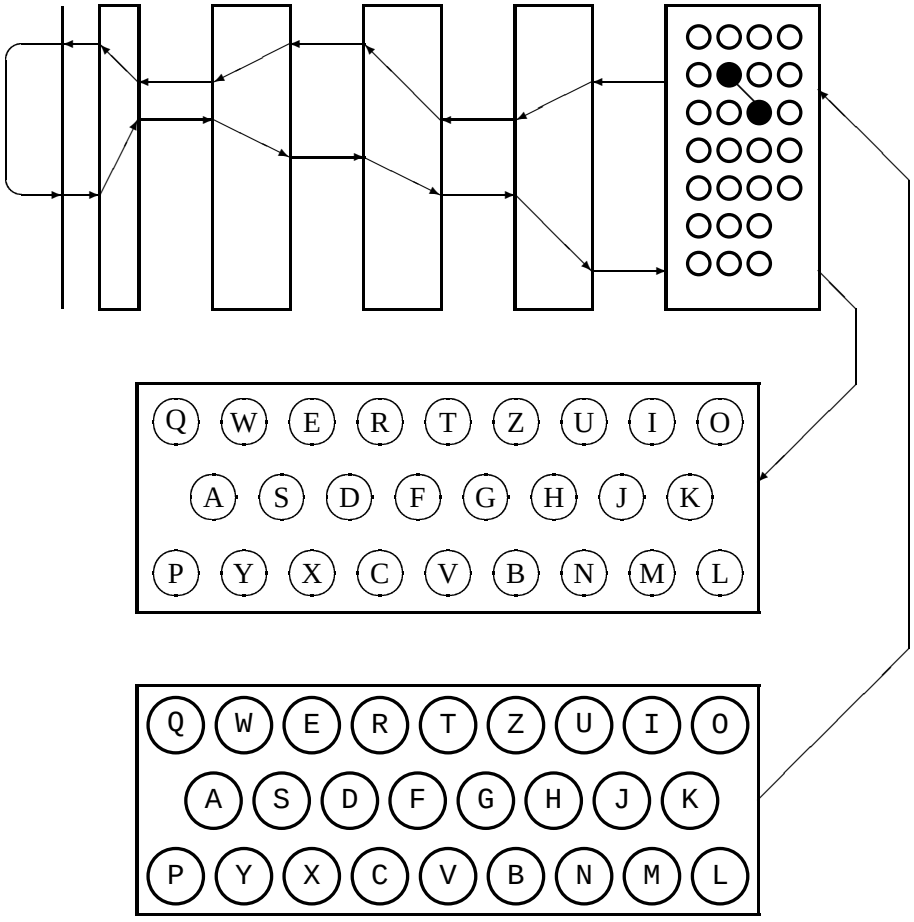
```
Dit is geheimschrift ...
Eju jt hfifjntdisjgu ...
```

Vanaf de 16de tot de 20ste eeuw gebruikten de verschillende Westerse mogendheden voornamelijk de zogenaamde Vigenère code. Feitelijk is dit niets anders dan de Caesar codering. Het enige verschil is dat letters op verschillende posities verschillend worden verschoven. Verschuiving met bijvoorbeeld het codewoord `bal` zorgt ervoor dat de eerste letter van een te coderen tekst één positie wordt opgeschoven (verschuiving over `B`), de tweede letter blijft onveranderd (verschuiving over `A`), en de derde letter wordt 11 posities verschoven (verschuiving over `L`). Daarna herhaalt zich dit weer. De vierde letter wordt verschoven met `B`, de vijfde over `A`, etc. Voor bovenstaande tekst leidt dit tot:

```
Dit is geheimschrift ...
Eie js rfhpjmddhcjfe ...
```

[1] Deze projectwijzer is gebaseerd op eerdere projectwijzers voor OGO2.1. In 2000-2001 was dit Ik zag twee beren brood sorteren. Experimenten met abstraheren, implementeren en evalueren. Door R. van Geldrop, T. Verhoeff en G. van Hoeve-Brouwer. In 2001-2002 ging de OGO2.1 opdracht over het positioneren van proteïnen in het menselijke genoom.

Al gedurende de 19de eeuw waren er methoden gepubliceerd om de Vigenère code systematisch te breken. Er ontstond behoefte aan een verbeterde encryptie-techniek die bovendien eenvoudig te gebruiken was. De vooruitgang in electro-techniek en mechanica leverde de ingredienten voor rotor machines. Zij bestaan uit een toetsenbord verbonden met één of meer rotoren. De rotoren draaien bij het aanslaan van een letter en iedere rotor voert een permutatie uit op de letters. In zekere zin is dit een Vigenère code – met letterpermutaties ipv. verschuivingen – waarbij de lengte van het codewoord wordt bepaald door het aantal draaingen nodig om de rotoren weer in de beginpositie te krijgen. De Amerikanen gebruikten de Hebern rotor machines. De Japanners hadden de op de Hebern machine geinspireerde RED cryptograaf. De rotormachine die Duitsers gebruikten heette de Enigma, waarvan er vele varianten bestonden. Alle Enigma's hebben 3 draaiende rotoren met elk 26 beginposities. Dat betekent dat het codewoord $26^3 = 17576$ tekens lang is. De standaard technieken om Vigenère codes te breken zijn daardoor niet toepasbaar op rotormachines.



Figuur 1: De Enigma zoals gebruikt door de Duitse marine

De Enigma die wij beschouwen is de meest gecompliceerde. Hij werd gebruikt door de Duitse marine aan het eind van de tweede wereldoorlog. Hij ziet er uit als in figuur 1.

De rotoren in de Enigma zijn verwisselbaar. De drie dikke rechter rotoren kunnen worden gekozen uit een set van 8. Voor de dunne linker rotor zijn een speciale beta en gamma rotor beschikbaar. Deze laatste rotor draait overigens niet tijdens het coderen. Helemaal aan het linkereind bevindt zich een reflector die het signaal terugstuurt. Hiervoor zijn eveneens twee typen beschikbaar.

Een letter wordt via een parallelle bus met 26 signaallijnen vanaf het toetsenbord naar een plugbord gestuurd. In dit plugbord kunnen door middel van stekkers letters omgewisseld worden. Voor de oorlog werden 6 stekkers gebruikt. Tijdens de oorlog werd het aantal opgevoerd naar 10. Wij zullen er één gebruiken. Daarna ging het signaal door de rotoren en via de reflector, de rotoren en opnieuw het plugbord naar een display. Het grote voordeel van de enigma is de symmetrie tussen codering en decodering. Een bericht dat gecodeerd is met een bepaalde setting, kan simpelweg worden gedecodeerd door het in te typen met dezelfde setting. Op deze wijze waren geen aparte codeer en decodeer machines nodig.

Ieder gecodeerd bericht dat werd verzonden werd voorafgegaan door drie letters. Dit gaf een nieuwe instelling van de drie rechter rotoren van de Enigma aan. De rest van het bericht werd met deze nieuwe instelling gedecodeerd. Tot 1939 verzonden de Duitsers de drie letter code zelfs tweemaal achterelkaar om bij enkele transmissiefouten verlies van het hele bericht te voorkomen. De hierdoor veroorzaakte zwakte was voldoende voor de Polen om alle gecodeerde berichten van de Duitsers te kunnen lezen. Wij beschouwen slechts teksten zonder voorafgaande drie letter code.

Hoeveel verschillende codeersleutels heeft de Enigma zoals wij die hier beschouwen? De rotors in de Enigma kunnen in $8 \cdot 7 \cdot 6 \cdot 2 \cdot 2$ verschillende plaatsen worden gepositioneerd. Bovendien kunnen alle rotoren in 26 beginposities worden gezet. Dit levert

$$8 \cdot 7 \cdot 6 \cdot 2 \cdot 2 \cdot 26^4 = 614\ 10^6$$

verschillende sleutels. De rotors bevatten een letterring die ook nog ten opzichte van de binnenring verschoven kan worden. Deze mogelijkheid zullen we buiten beschouwing laten.

Het plugbord met p pluggen multipliceert het aantal sleutels nog met een factor

$$\frac{26!}{2^p\, p!(26-2p)!}.$$

Hoewel wij slechts één plug gebruiken, zijn er ruim drie weken nodig om alle mogelijke teksten te decoderen en te doorzoeken, aannemende dat er 10^5 teksten per seconde kunnen worden vertaald en doorzocht. Er is ten hoogste 24 uur beschikbaar.

Om deze rekenklus uit te voeren wordt de "*Unite*" (CRAY Origin2000) supercomputer ingezet. Deze computer bestaat uit 128 processoren en een geheugen van

56 Gbyte. Hij is uitgerust met speciale bussen om communicatie tussen de verschillende processoren en communicatie met IO-devices extra snel te laten verlopen. De kracht van de Unite zit met name in zijn geheugen, aantallen processoren, 64 bits architectuur en snelle bussen. De individuele processoren zijn inmiddels relatief langzaam. De Universiteiten van Twente en Eindhoven hebben de Unite samen bekostigd. Hij wordt beheerd door het nationale rekencentrum SARA in Amsterdam (`www.sara.nl`). De Unite wordt voor dit OGO 2.1 project beschikbaar gesteld gedurende een korte periode aan het eind van het eerste blok, en gedurende een korte periode aan het eind van het tweede blok.

Gedurende het project wordt in groepen van 5 à 6 personen samengewerkt. De samenstelling van de groepen wordt bij de eerste bijeenkomst bekendgemaakt. Binnen de groepen moet worden bepaald welke taken er zijn, hoe ze verdeeld moeten worden en wanneer de taken moeten worden afgerond. Er is wekelijks 8 uur voorzien voor de uitvoering van dit project. Daarvan kan de groep gedurende 4 uur per week gebruik maken van ingeroosterde zalen. Er worden aan het begin een speciaal op OGO2.1 gerichte colleges gegeven, en er volgt aan het eind een eindpresentatie. Aan het begin van het project wordt een korte vergadertraining gehouden waarover informatie volgt.

Begeleiding vindt plaats door studenttutoren, die vooral letten op het goed verlopen van het groepsproces. De tutoren zelf hebben geen bijzondere expertise op het gebied van codering en supercomputers en mogen geen voortrekkende rol in de groep of het groepsproces spelen. Wel kunnen ze hun ervaring en expertise inzetten om het proces bij te sturen, door bijvoorbeeld te verwijzen naar ter zake kundige personen of literatuur, of te helpen voor en nadelen van verschillende werkwijzen goed tegen elkaar af te zetten. In het geval er binnen een groep problemen optreden die door de studenttutoren niet goed kunnen worden opgelost, kan altijd contact opgenomen worden met de projectcoördinator.

Het project wordt door de groepen zelf opgedeeld en gepland. Dit werkplan dient bij de tutor ingeleverd te worden. Indien het werkplan tussentijds een aanpassing behoeft (en dat is een normale gang van zaken!), dient de groep deze aanpassingen aan de tutor te verstrekken.

Aan het eind van het eerste blok wordt de Unite beschikbaar gesteld om een door middel van Vigenère gecodeerde tekst te decrypten. Hierover hoeft geen verslaglegging plaats te vinden. De gedecrypte tekst moet aan tutoren en projectcoördinator beschikbaar worden gesteld. Het doel van deze decodering is om ervaring op te doen met de Unite. In principe heeft het decoderen van deze tekst geen invloed op het eindcijfer.

Aan het eind van het tweede blok moet een door de Enigma gecodeerde tekst worden gedecodeerd. Het conceptverslag wordt door de projectcoördinator bekeken om te bepalen of groepen zich voldoende hebben voorbereid om met redelijke kans de tekst binnen 24 uur te kunnen decoderen. Alleen deze groepen krijgen voor de tweede maal toegang tot de Unite.

De beoordeling van dit project is gelijk verdeeld over enerzijds het decodeerprogramma en anderzijds verslag en proces.

De beoordeling van het programma is als volgt. Programma's die de tekst fout decoderen worden met een 3 beoordeeld. Deze beoordeling geldt ook voor programma's die niet op de Unite mogen worden uitgevoerd. Programmas die niet in staat zijn tot het decoderen van de tekst in 24 uur of waarvan het programma termineert zonder het opleveren van de correcte gecodeerde tekst worden met een 4 beoordeeld. Beide beoordelingen kunnen in een 6 worden omgezet wanneer aannemelijk kan worden gemaakt dat dit onvoldoende resultaat verkregen is op basis van redelijke en in het verslag beschreven aannamen die niet van toepassing bleken op de te decoderen tekst. Het programma dat in staat is met de minste rekentijd de tekst correct te decoderen wordt met een 10 beoordeeld. De tijd die in beschouwing genomen wordt is de tijd vanaf het starten tot terminatie van het programma op de Unite. Het gebruikte aantal processoren wordt hierbij buiten beschouwing gelaten. Goed gebruikt parallellisme is dus een voordeel.

Het (concept)verslag wordt beoordeeld op zijn technisch merites: kan een lezer efficient begrijpen welke technieken zijn gebruikt en welke resultaten zijn behaald. Hierbij hoort een adequate structurering, duidelijke probleemstelling, precisie, voorbeelden. Een goed verslag is compleet, begrijpelijk en beknopt. Het verslag moet minimaal een Engelse samenvatting bevatten en mag in zijn geheel in het Engels geschreven worden. De coördinator bepaalt het eindcijfer dat voornamelijk zal zijn gebasseerd op het verslag, maar dat ook beinvloed wordt door groepsproces en indien van toepassing de eindpresentatie.

Bij de eindpresentatie zullen enkele geselecteerde groepen een presentatie geven. Dit wordt pas tijdens of vlak voor de eindpresentatie bekend gemaakt. Alle groepen zullen dus een eindpresentatie moeten voorbereiden.

Om verschillen in een groep te verdisconteren in het eindcijfer zullen groepsleden elkaar anoniem beoordelen. Dit gebeurt enerzijds tussentijds waarmee ieder groepslid zijn eigen positie beter leert begrijpen. Dit gebeurt anderzijds aan het eind waarmee de projectcoördinator het groepscijfer zal verfijnen tot individuele cijfers.

Er is een webpagina voor OGO 2.1:

```
www.win.tue.nl/~jfg/educ/2R660/2002.html
```

Daar wordt aanvullende informatie bij deze projectwijzer gegeven.

2 De projectopdrachten

2.1 Opdracht 1: Vigenère

Op de Unite staat in de directory

```
/tue/ewin/ewsjfg/vigenere
```

in de file `opdracht1.txt` een door middel van Vigenère gecodeerde tekst. Deze tekst is op een grafsteen in Indonesie gevonden en stamt uit ongeveer 1900. De

tekst bestaat uit hoofd en/of kleine letters, spaties en controlcodes ten behoeve van regelovergangen (hiervoor worden op verschillende systemen verschillende controlcodes gebruikt!). De spaties en controlcodes zijn er slechts ten behoeve van de layout. De tekst bevat minder dan 1024 letters.

Op donderdag 26 september worden passwords uitgereikt, en kan het programma worden uitgevoerd. Na enkele dagen zullen de accounts weer worden dichtgezet.

2.2 Opdracht 2: De Enigma

Zoals eerder uitgelegd bestaat de Enigma uit drie posities voor dikke rotoren en één voor een dunne. Er zijn acht dikke rotoren die de volgende permutatie uitvoeren:

```
rotor[1]="EKMFLGDQVZNTOWYHXUSPAIBRCJ";
rotor[2]="AJDKSIRUXBLHWTMCQGZNPYFVOE";
rotor[3]="BDFHJLCPRTXVZNYEIWGAKMUSQO";
rotor[4]="ESOVPZJAYQUIRHXLNFTGKDCMWB";
rotor[5]="VZBRGITYUPSDNHLXAWMJQOFECK";
rotor[6]="JPGVOUMFYQBENHZRDKASXLICTW";
rotor[7]="NZJHGRCXMYSWBOUFAIVLPEKQDT";
rotor[8]="FKQHTLXOCBJSPDZRAMEWNIUYGV";
```

Dus rotor 0 vertaalt een A in een E, mits hij nog niet gedraaid is (dwz. in positie A staat). Op de terugweg wordt een E weer in een A vertaald. De twee dunne rotoren zijn alsvolgt bedraad:

```
rotor[9]="LEYJVCNIXWPBQMDRTAKZGFUHOS"; /* beta  */
rotor[10]="FSOKANUERHMBTIYCWLQPZXVGJD"; /* gamma */
```

De dunne reflectoren voeren de volgende permutaties uit:

```
thinreflector[1]="ENKQAUYWJICOPBLMDXZVFTHRGS";
thinreflector[2]="RDOBJNTKVEHMLFCWZAXGYIPSUQ";
```

Om te begrijpen hoe de middelste en linker rotor bewegen, moeten we naar het ‘carry’ mechanisme kijken. Wanneer een rotor bij de ‘notch’ positie komt, steekt een staafje in de linker rotor. Bij de volgende toetsaanslag duwt een speciale stang tegen het staafje, en duwt beide rotoren een positie verder. De beta en gamma rotoren draaien niet. Op

```
http://homepages.tesco.net/~andycarlson/enigma/enigma_j.html
```

is een enigma simulator te vinden die dit op elegante wijze illustreert. De staafjes worden uitgestoken op de hieronder aangegeven posities. Drie rotoren hebben twee notches.

```
notch[1]="Q";
notch[2]="E";
notch[3]="V";
notch[4]="J";
notch[5]="Z";
notch[6]="ZM";
notch[7]="ZM";
notch[8]="ZM";
```

Onderstaande tekst is gecodeerd met rotors beta, 4, 2 en 3 en de eerste dunne reflector. De beginsetting voor de rotoren was B, R, F en D. Op het plugboard waren C en E verbonden. Het was gebruikelijk om een X voor een spatie te schrijven.

```
anXde nXcom manda ntXof fizie rXher manXk lausX wette rXvor
hersa geXmi ttagX ueber wiege ndXvo nXzei tweis enXau fheit
erung enXab geseh enXst aerke Xrbew oelkt XundX immer Xwied
erXre genXo derXs chaue rXoer tlich Xauch Xgewi tterX abXun
dXzuX naech stXno chtei lsXwo lkigX aberi Xmeis tXtro ckenX
spaet erXso garXg ebiet sweis eXwol kenlo s
```

De vertaling hiervan luidt.

```
RTEWV LNOTB FKPFP PQWTR MZHVT KJPYL HPVFB BXRYJ PMBXM WARGC
EAMRG IDTXD JGVWB VAAPG YGQVM KSITM VJSNE QTDMI GQJEA VPFEA
SEVBK UGIPU CFNYS AYBWV BPTVD KUPQB CVQWY EVOZR YNINQ QACXL
GLHVI ZJBVM JAXJB SWOPQ PMRWM XJARX JPGFZ LEXCP NOTCJ XOJLA
QGDWL FWGKO ZJXHY DZQCN HLEQF IWXDB WWCQJ ALFWZ CUXBR ZKIZX
HFHLQ FKAMU TGRUG QOJNJ AYTHP LSLPS LKBQB U
```

De inputfile bevindt zich in de directory

```
/tue/ewin/ewsjfg/enigma
```

in de file `opdracht2.txt`. Vanwege het moment van verzending van het bericht is het aannemelijk dat het bericht eveneens informatie over het weer bevat. Omdat de duikboot van Herman Klaus echter al geruime tijd zonder bevoorrading op zee is geweest is het ook mogelijk dat het werkwoord ‘treffen’, of ‘der Treffpunkt’ in de tekst voorkomt. Tot ieders verbazing heeft ‘Gefreiter’ Schnur recentelijk in code gevraagd of hij vader van een zoon of dochter geworden is. Soms wordt zo’n vraag aan de vragensteller gericht beantwoord. Overigens is het voorschrift om een bericht met een willekeurig woord of letters te laten beginnen. Zoals boven blijkt gebeurt dat niet altijd.

De programmatuur en de file met het resultaat moet op de Unite achterblijven. Verder moet het resultaat worden opgestuurd naar de studentassistent voor de programmeertaal en Unite begeleiding en de tutor.

3 Aanwijzingen en achtergrondinformatie

3.1 Algoritmes en probleemanalyse

Bij deze opdracht wordt gezocht naar een programma dat snel en correct werkt. De ervaring leert dat programmatuur het best geconstrueerd kan worden door eerst een goede analyse te maken van het probleem en de structuur ervan goed te doorzien. In het bijzonder geldt voor dit decodeer probleem dat de meeste performancewinst behaald wordt uit een mathematische analyse van de werking van de Enigma. Door vast te stellen dat de resulterende tekst uit aanvaardbaar Duits bestaat, kan bepaald worden of de decodeerinstelling goed of fout is. Omdat ook dit waarschijnlijk vaak moet worden uitgevoerd, is het effectief ook hier een efficient algoritme voor te verzinnen.

De tweede stap bestaat uit het opstellen van schema waarbij de layout van het programma in grote stappen wordt gekarakteriseerd. Probeer zo precies mogelijk de eigenschappen vast te leggen van de datastructuren in de tussenfases! Gebruik formules omdat daarmee een voldoende precisie bereikt wordt. In de derde fase kunnen pseudo algoritmes worden opgesteld om de verschillende stappen in te vullen.

Aarzel niet om algoritmes uit de literatuur te gebruiken en vermeld daarbij altijd precies de bron op de daarvoor vastgelegde manier. Over de Enigma is erg veel bekend. Er zijn hele boeken over geschreven. De meeste leerboeken over cryptologie bevatten een hoofdstuk over rotor gebaseerde codeermachines met aanknopingspunten naar verdere literatuur. Voor meer specialistische aspecten zullen vaktijdschriften zoals *Cryptologia* geraadpleegd moeten worden. Onze bibliotheek heeft een uitstekende collectie boeken en tijdschriften. Ook het Internet is een bron met zeer veel informatie.

Reken uit hoeveel ruimte en tijd de algoritmes zullen vergen, bij de gegeven omvang van de input. Dit maakt het in dit stadium mogelijk te bepalen waar de performance bottlenecks zitten, die alvast verholpen kunnen worden.

Pas als alle voorgaande stappen zijn gezet, en iedereen er van is overtuigd dat dit algoritme correct en voldoende snel is, kan de pseudocode worden omgezet naar een werkend programma. Vergeet niet dit programma te testen, om eenvoudige verschrijvingen te detecteren, en om nogmaals overtuigd te raken van de correctheid van het programma.

Een veel gebruikte methode om software te maken bestaat uit het zonder vorm van à priori analyse coderen van een prototype, dat daarna net zolang wordt bewerkt totdat “de gewenste output” voldoende snel wordt verkregen. Hoewel deze wijze van werken relatief snel resultaat oplevert, leidt zij vaak tot programmatuur die te vaak toch niet goed werkt, en waarvan de documentatie, indien hij al geschreven wordt, meestal niet een goede, compacte en complete reflectie is van de vaak ondoorzichtige software. In de praktijk leidt deze werkwijze vaak tot lage initiele kosten en tot snel succes van eenvoudige projecten. Bij ingewikkelder projecten blijken bij deze werkwijze vaak dat het product alsmaar ‘bugs’ te bevatten,

die zich slechts met relatief grote inspanning laten verwijderen. Deze 'bugs' zijn vaak een uiting van een incorrecte basisstructuur. Een dergelijk project is alleen te redden door het helemaal opnieuw op te zetten.

3.2 De Cray Origin2000: Unite

Gegevens over de Unite zijn te vinden op `www.sara.nl` en `www.sgi.com`. Hier zijn ook datasheets te vinden. In de tweede week zal een korte presentatie over de Cray Origin2000 gegeven worden. De Cray zal tweemaal enige tijd beschikbaar worden gesteld voor dit project. Preciese data staan achter in deze wijzer. Tzt. zullen door de tutoren toegangscodes en passwords voor deze machine beschikbaar worden gesteld.

De Cray beschikt over ca. 56GB geheugen en 128 processoren. Iedere processor benadert het geheugen dmv. caches. Iedere processor heeft 4MB second level cache tot zijn beschikking en direct toegang tot ca. 1GB main memory. Logisch gesproken heeft iedere processor toegang tot al het beschikbare geheugen en een programmeur hoeft in principe geen rekening te houden met het fysieke geheugenmodel. Toch is het goed rekening te houden met het feit dat er een verschil van een factor 100 zit tussen de access tijden van data die dichtbij in de cache zit, en data die van ver moet komen.

De processoren van CRAY zijn van het type R10000. Deze processoren zijn niet erg snel. De kracht van deze computer zit vooral in de aanwezigheid van snelle databussen en de beschikbaarheid van veel geheugen en veel processoren.

3.3 IRIX operating systeem

De Cray Origin2000 gebruikt een van Unix afgeleid operating systeem dat Irix heet. Om ervaring op te doen met het Irix operating systeem wordt de

```
sgtt08.win.tue.nl
```

beschikbaar gesteld. Dit is een kleine desktop computer die het IRIX operating systeem draait. Doe hierop geen onnodige experimenten om andere gebruikers niet in de weg te zitten. Toegangscodes en passwords voor deze machine zullen bij BCF beschikbaar zijn.

De belangrijkste commando's van IRIX worden hier gegeven. Van iedere commando is meer informatie op te vragen door het commando `man`. Type bijvoorbeeld `man ls` om meer over het commando ls te weten te komen. Hier staan ook vaak verwijzingen naar gelijksoortige commando's.

`ls`	Geeft inhoud van een directory
`cd`	Verander de directory
`mkdir`	Maak een nieuwe directory
`pwd`	Geef het pad naar de huidige directory
`chmod`	Verander toegangspermissies van een file of een directory

`vi`	Edit een file
`more`	Bekijk een file
`less`	Bekijk een file, geavanceerder dan `more`
`rlogin`	Login op een andere machine. Met de vlag `-l user` kan onder een andere naam worden ingelogd.
`ssh`	Zelfde als rlogin, maar nu dmv. een secure protocol
`telnet`	Zelfde als rlogin.
`ftp`	Kan worden gebruikt om files tussen computers te transporteren
`scp`	Zelfde als `ftp`; maakt ook gebruik van een secure protocol
`qsub`	Submit een batch job om te worden ge-executeerd (alleen Unite)
`limit`	Geeft de beperkingen op resources voor interactieve jobs
`jobinfo`	Geeft informatie over batchjobs
`cc`	De SGI C compiler
`gcc`	De Gnu C compiler (alleen 32 bits, dwz. max 2GB geheugen toegankelijk)
`gdb`	De Gnu debugger
`dbx`	De SGI debugger

De programma's `ftp` en `telnet` zijn ook op de pc beschikbaar.

3.4 Programmeertaal

De keuze voor een programmeertaal is vrij, maar omdat de Unite voornamelijk C en C++ ondersteunt zijn ligt het gebruik van deze talen voor de hand. C lijkt erg op Pascal, maar is veel verraderlijker, omdat het niet erg strikte typechecking kent. Een bekend voorbeeld is `if (x=1) {...} else {...}`. De schijnbare conditie `(x=1)` is een assignment. Het resultaat van het assignment is de waarde van de rechterkant. Indien `x` oorspronkelijk gelijk was aan `0`, dan zal na afloop `x` gelijk zijn geworden aan `1`. Het resultaat van de vergelijking is eveneens `1`, en die staat voor de boolean true, en dus wordt de then tak uitgevoerd. De beoogde check had geschreven moeten worden als `(x==1)`. Dit betekent dat C programma's met een extra grote zorgvuldigheid moeten worden geschreven.

Er wordt vanuitgegaan dat iedere groep zich zelf de taal C eigen maakt (zie bv. [3]). Ter ondersteuning is er op de website een artikel over C voor Pascalprogrammeurs van Jeroen Fokker gezet en is er een programmeertaal-expert beschikbaar die ondersteuning kan bieden. Hier bespreken we slechts de compilatie van C programma's binnen een Unite omgeving.

Het belangrijkste commando is `cc`. Dit is de C compiler voor IRIX. De Gnu C compiler is ook beschikbaar, en werkt vrijwel hetzelfde. Een file kan worden gecompileerd door het commando `cc -o file file.c`. Als dit commando succesvol is uitgevoerd staat de gecompileerde code van `file.c` in `file`. Door `file` (of soms `./file`) kan de code worden uitgevoerd. Bij het `cc` commando kunnen vlaggen worden meegegeven. Bv. `cc -g -O2 -o file file.c`. De vlag `-g` betekent dat debug informatie meegegeven moet worden opdat `dbx` of

`gdb` interne programmadata aan de variabelen van het programma kan verbinden. De vlag `-O2` geeft aan dat de compiler moet optimaliseren, wat al snel tientallen procenten snelheidswinst kan opleveren. De vlag `-fullwarn` (of `-Wall` voor `gcc`) zorgt ervoor dat de compiler mogelijke problemen in de code aangeeft. Met de vlag `-64` wordt 64 bits code gegenereerd, nodig wanneer er meer dan 2GB geheugen nodig is voor het programma. Deze vlag zorgt er voor dat pointers uit 8 bytes bestaan, ipv. 4. Het geheugenbeslag neemt dus met een factor 2 toe. De vlag `-64` werkt niet voor de `gcc` compiler.

3.5 De programmarun op de Unite

Het programma moet op de Unite in batchmode worden uitgevoerd. Op deze manier kan gebruik worden gemaakt van de resources van de Unite. Bovendien worden de runs van alle batchprogramma's gelogd, en kan op deze manier op uniforme wijze worden nagegaan hoeveel tijd de programma's van de verschillende groepen gebruiken. De Unite staat het ook toe om interactief te werken, maar dit gebruik is sterk gelimiteerd, zodat het eigenlijk alleen geschikt is om programma's te compileren en een eenvoudige testrun uit te voeren.

Een job wordt gesubmit door het commando `qsub batchfile`. In de batchfile staan gegevens over het te verwachten gebruik van resources. Zuiniger programma's zullen eerder worden uitgevoerd. Een voorbeeld van een `batchfile` is:

```
#!/bin/csh
#QSUB -lT 1:00:00       # totale geschatte CPU tijd
#QSUB -lM 10Gb          # geschat geheugengebruik
#QSUB -l mpp_p=1        # aantal benodigde processoren
#QSUB -re -ro
#QSUB -e job_test.error
#QSUB -o job_test.qout
setenv PATH $QSUB_PATH
<naam van programma>
```

Voor `QSUB_PATH` moet het pad naar de directory worden gezet waar de executables zich bevinden. De file `batchfile` moet executiepermissie en leespermissie hebben. Submit als proef eerst een kleine batchjob om te kijken of alle instellingen in orde zijn. Indien er gebruik gemaakt moet worden van veel processoren of geheugen, zoek dan uit of dit wel is toegestaan. In principe mogen er ten hoogste 20GB geheugen en 64 processoren worden gebruikt.

Een veel voorkomende fout is om bovenstaande batchfile op een PC te maken. Het end-of-line character op de PC en de Unite verschilt. Het programma dat de batches verwerkt raakt hopeloos in de war van het end-of-line character van de PC en produceert dan onnavolgbare uitvoer.

Na het submitten van het programma moeten alle programma-, resultaat- en outputfiles ongewijzigd op de Unite blijven staan, opdat controle mogelijk is, en

Start project, voordracht over planning	dinsdag 3 september, 3-4de uur
Voordrachten over Unite en bibliotheek	dinsdag 10 september, 3-4de uur
Uitvoeren van opdracht 1 op de Unite	donderdag 26 september
Wederzijdse tussenbeoordeling	donderdag, 3 oktober
Inleveren conceptverslag	donderdag 24 oktober
Uitslag toegang tot Unite	donderdag 31 oktober
Uitvoeren opdracht 2 op Unite	donderdag 7 november
Presentatie en groepsvoordracht	dinsdag 12 november
Wederzijdse eindbeoordeling, eindbeoordeling en nabespreking	donderdag 14 november

Tabel 1: Belangrijke data

opdat, indien nodig, het programma nogmaals kan worden gesubmit door de begeleiding. In het eindverslag moeten duidelijk de namen van de verschillende files worden aangegeven en de resultaten van de runs worden vermeld. Om deze resultaten te controleren zal het mogelijk zijn nog enige tijd op de Unite in te loggen.

4 Fasering en planning

De planning en taakverdeling van dit project liggen geheel in handen van de groep. De planning zal opgehangen moeten worden aan de belangrijke data genoemd in tabel 1. De planning en de taakverdeling, alsmede wijzingen hierop gedurende het project, moeten bij de tutor worden ingeleverd.

Het is echter belangrijk bij de tijdsplanning rekening te houden met het feit dat de deadlines strikt zijn. Het is daarom van belang defensief te plannen, d.w.z. met voldoende ruimte om onvoorziene omstandigheden op te vangen. Het is ook van belang snel (dwz. in de eerste week) tot een taakverdeling te komen. Het is niet nodig dat iedereen alles doet, maar het is wel zo dat iedereen in de groep verantwoordelijkheid voor het geheel draagt. Besteed uitgebreid aandacht aan hoe te garanderen dat het programma correct en snel genoeg is en niet teveel geheugen gebruikt. Belangrijke technieken hierbij zijn mathematische modellen van datastructuren en complexiteitsanalyse. Er is niets tegen om gebruik te maken van de kennis van de tutor. Het binnen de groep uitleggen van de preciese werking van het (beoogde) programma werkt ook zeer heilzaam. Code inspectie door anderen dan de programmeur is een van de erkende zeer effectieve hulpmiddelen om programma's correct te maken, zeker wanneer dit ondersteund wordt door gebruik van asserties en invarianten in de code. Schuw het uitvoeren van testen niet, maar houd er rekening mee dat hiermee meestal alleen oppervlakkige fouten worden gevonden.

5 Groepen

De groepen bestaan elk uit 5 à 6 studenten. Ze worden willekeurig samengesteld. Tijdens de eerste bijeenkomst wordt de groepsindeling bekendgemaakt en wie er als tutor op zal treden.

6 Overleg

Op de volgende wijze kan er met de projectleiding overlegd worden over voortgang van het project.

Projectgroepoverleg Dit is het wekelijkse overleg dat leden van een projectgroep met hun tutor hebben. Onderwerp is de inhoud en voortgang van het project. Het overleg duurt per groep ca. een half uur. Voor de groepen 1-6 zal het projectgroepsoverleg op donderdagochtend in de voor OGO2.1 gereserveerde zaal plaatsvinden (zie hiertoe het zalenrooster). Voor groepen 7-12 vindt het projectgroepsoverleg op donderdagmiddag plaats.

Projectcoördinator De projectcoördinator kan worden benaderd in het geval van problemen die om welke reden dan ook niet door de tutor kan worden afgehandeld. Dit kan per mail, of via een afspraak met de secretaresse T. van de Bosch, tel. 040-247 5010, `wsintech@win.tue.nl`.

7 Begeleiders

De volgende begeleiders zijn betrokken bij dit project:

1. De tutoren zijn J.H.J. Kennes (Judith), T.H.M. (Thijs) Janssen, M.A. (Maurice) Termeer, R. (Ronald) Kruidhof. Alle zijn te bereiken via (Voorletter.)*.Naam@student.tue.nl.

2. De projectcoördinator is J.F. Groote (J.F.Groote@tue.nl, HG6.75, tel. 5003/5010).

3. De opleidingsonderwijskundige is J.C. Perrenet (J.C.Perrenet@tue.nl, HG6.34, tel. 3439). De taak van de onderwijskundige is het bewaken van de grote lijn in OGO.

4. M.J. (Mike) Holenderski (M.J.Holenderski@student.tue.nl) is een studentassistent die helpt bij problemen inzake de te gebruiken programmeertaal en het gebruik van de Unite. Hij kan benaderd worden met implementatievragen. Er zal een wekelijks spreekuur zijn van 10.00 tot 12.00 op donderdagen. Hij zal zich bevinden in HG6.73.

8 Producten en documenten

8.1 Werkplan

In de eerste week moet een werkplan worden opgesteld, die moet worden ingeleverd bij de tutor. Het werkplan omvat een taakverdeling voor de individuele leden van de groep. Wijzigingen op het werkplan moeten eveneens worden ingeleverd.

8.2 Logboeken

Iedere groep moet bijhouden hoeveel tijd hij aan welke activiteit heeft besteed. Deze logboeken moeten op de laatste bijeenkomst bij de tutor worden ingeleverd. De logboeken hebben twee functies. Ten eerste dienen ze om inzicht te krijgen in de tijdsbesteding van iedere betrokkene bij dit project om te toetsen of de planning realistisch was. Ten tweede dienen ze voor de coördinator als terugkoppeling op het project, zodat die indien nodig volgend jaar bijgesteld kan worden.

8.3 Conceptverslag

Om toegang te verkrijgen tot de Unite moet een conceptverslag (afgedrukt op papier) worden ingeleverd dat een beschrijving van het te runnen programma bevat, analyses van correctheid en complexiteit. Het te runnen voldoende geannoteerde programma moet ook worden meegeleverd (mag electronisch). Het conceptverslag mag in het Engels of het Nederlands worden geschreven, maar moet een Engelse samenvatting bevatten. Zorg dat het conceptverslag zo kort mogelijk is (< 10 pagina's bij voorkeur), maar alle technisch van belang zijnde overwegingen bevat. Het verslag bevat dus wel afwegingen tussen verschillende algoritmes, en typisch niet een evaluatie van groepsprocessen oid. Een goede maatstaf is dat iemand die niet inhoudelijk bij het project betrokken is, er na lezing van het verslag begrijpt waarom voor zekere aanpak is gekozen en in staat is het project voort te zetten.

8.4 Eindverslag

Het eindprodukt bestaat uit het eindverslag dat is gebaseerd op het conceptverslag waarin de uitkomst van de run op de Unite vermeld staan, en waarin aanwijzingen gemaakt op het conceptverslag zijn verwerkt. Het eindverslag wordt op papier ingeleverd. Het programma en de output moeten electronisch worden ingeleverd, en moeten beschikbaar zijn op de Unite.

8.5 Presentaties

Er zal een eindpresentatie van 20 minuten gegeven worden in het Nederlands door geselecteerde groepen tijdens de eindbijeenkomst. Pas ter plekke wordt bekend gemaakt welke groepen een voordracht zullen geven. Deze eindpresentatie moet ter oefening al aan de groepsleden in het bijzijn van de tutor gegeven zijn.

9 De eindbeoordeling

Het projectwerk verricht door de groep wordt primair beoordeeld door de projectcoördinator in overleg met de tutor op grond van het eindverslag en de kwaliteit van het programma. De cijferbeoordeling staat aan het begin van dit document. Eindresultaat enerzijds en eindrapportage en groepsproces vormen beide de helft van het eindcijfer.

Om mee te nemen dat er binnen groepen soms een verschil in inzet bestaat, moeten de groepsleden elkaar anoniem beoordelen met een $++$ (is een voortrekker), $+$ (is een nuttig lid van de groep), $\pm$ (vormt een neutraal lid van de groep), $-$ (zet zich matig in), $--$ (maakt zich er van af). De individuele cijfers kunnen met één punt (en in extreme gevallen met twee punten) omhoog of omlaag worden bijgesteld op basis van deze uitkomsten. Halverwege wordt al een onderlinge anonieme proefbeoordeling gegeven. Deze heeft geen invloed op de eindbeoordeling. Maar mede op basis van deze proefbeoordeling kan door de projectcoördinator besloten worden tot uitzetting van negatief beoordeelde groepsleden.

Door enkele geselecteerde groepen zullen eindpresentaties worden gegeven. Deze zullen niet worden meegenomen in de eindbeoordeling, maar dienen om medestudenten en begeleiders op de hoogte te stellen van de aanpak. Tijdens de eindbijeenkomst wordt bekend gemaakt wie een presentatie zal geven.

Na de eindbijeenkomst volgt nog een bijeenkomst met de tutor. Hij zal het verslag, eindresultaat en het groepsproces doornemen.

Referenties

[1] C.A. Deavours en L. Kruh. Machine cryptography and modern cryptanalysis. Artech House, Dedham, 1985.

[2] J. Fokker. C voor Pascal programmeurs. Handout 1992.

[3] B.W. Kernighan en D.M. Ritchie. The C programming language. Prentice Hall 1988.

[4] A.G. Konheim. Cryptography: A primer. John Wiley and Sons, New York, 1981.

10 Tutorenwijzer

1. De primaire taak van de tutor is het groepsproces op gang te houden en te bewaken, en er voor te waken dat men snel van start gaat. Daartoe moet de planning in de tweede week worden ingeleverd. De tutoren en de programmadeskundige worden geacht het eerste college (dinsdag 3 september 3-4de uur) aanwezig te zijn om met de groep kennis te maken.

2. De opdracht is er op gericht om te denken over algoritmiek en om over correctheid en performance na te denken, zonder in staat te zijn de programma's in de volle omvang te runnen. De tutor moet de studenten wijzen op verschillende alternatieve aanpakken, maar moet ze vrij laten in de eigen keuze. De studenten hebben niet altijd de benodigde kennis en abstractievermogen en moeten in beide door de tutor op weg geholpen worden.

3. De onderlinge tussenbeoordeling moet worden gehouden en er moet voor gezorgd worden dat groepsleden niet van elkaar weten hoe ze elkaar beoordeeld hebben. Het is aan de tutor vast te stellen of deze beoordeling consequenties heeft voor zekere leden van de groep. In samenspraak met de coördinator kunnen maatregelen worden vastgesteld. De tutor heeft als taak de groep te beschermen tegen onwillige of onkundige leden. Studenten beoordelen ook zichzelf. Deze individuele beoordeling moet worden gemailed aan Jacob Perennet, die voor de beoordeling een speciaal formulier in excell heeft gemaakt die op de webpagina te vinden is. In de eindbeoordeling telt de onderlinge beoordeling alsvolgt mee. Voor $--$ krijgt men 0 punten, voor $-$ 1 punt, voor $\pm$ 2 punten, voor $+$ 3 punten, en voor $++$ 4 punten. Het cijfer van een individu wordt bepaald door de volgende formule:

   $$c_i = c_g + \frac{gem_i - gem_g}{2}$$

   afgerond op de normale wijze; c_g is de groepsbeoordeling. gem_i is de gemiddelde onderlinge beoordeling van een individu, en gem_g is de gemiddelde onderlinge beoordeling van een groep. c_i is het eindcijfer van een individu.

4. Iedere week, muv. de tussenweek vindt een overleg tussen coördinator en tutoren plaats om gang van zaken te bespreken. Het tijdstip waarop dit zal gebeuren moet nog nader bepaald worden.

5. Het conceptverslag wordt bekeken door de tutor en samen met de coördinator wordt bepaald of studenten toegelaten kunnen worden. Merk op dat voor de beoordeling weinig tijd is ingeruimd. Het eindverslag wordt beoordeeld door de coördinator.

6. Om iedereen te laten oefenen met voordrachten, is voor iedereen de kans ingebouwd dat ze een voordracht moeten houden. Ongeveer drie voordrachten zullen voor de hele groep worden gehouden.

7. De Unite deskundige levert na afloop van de experimenten een tabel waarop per groep staat hoeveel tijd ze gebruikt hebben, en hoe goed de zoekresultaten van de groepen waren.

8. Na afloop bespreekt de tutor met zijn groep zijn bevindingen. Er wordt door Jacob Perrenet een checklijst beschikbaar gesteld aan de hand waarvan de tutor het groepsproces door kan nemen.