Projectwijzer OGO 2.1 (2I040)
Het ontwerp van een dienstregeling voor de NS

Projectcoördinatoren: M. de Berg en J.F. Groote

Opleiding technische Informatica 2003–2004

Inhoudsopgave

De NS is bezig het spoornetwerk op een aantal plaatsen uit te breiden—verdubbelen van sommige lijnen van twee naar vier sporen, aanleggen van nieuwe lijnen, etc.—om daarmee de reistijden van de klanten reduceren. Om een reductie in de reistijden te kunnen bewerkstelligen is er natuurlijk een nieuwe dienstregeling nodig, die optimaal gebruik maakt van de uitbreidingen. Het doel van dit project is om software te ontwikkelen die de NS hierbij kan ondersteunen. Concreet betekent dit dat uw software uit een beschrijving van een spoornetwerk en data over aantallen reizigersbewegingen een zo goed mogelijke dienstregeling moet berekenen. Daarnaast moet de software, gegeven een dienstregeling en de reizigersbewegingen, de gemiddelde reistijd berekenen.

Het genereren van een geschikte dienstregeling voor een gegeven spoornetwerk is een bijzonder lastig probleem. Daarom zullen we een aantal aspecten niet in ogenschouw nemen in deze opdracht. Zo moet bij het opstellen van de dienstregeling eigenlijk ook rekening worden gehouden met de beschikbaarheid van personeel. Daarnaast moet er niet alleen gepland worden op welke tijdstippen treinen op een traject rijden, maar ook wat voor soort en hoeveel treinstellen ingezet worden. Zelfs met de vereenvoudigingen die we zullen doen blijft het probleem reken-intensief. Daarom zullen we de *Teras* supercomputer inzetten. Deze computer bestaat uit 1024 processoren en heeft een geheugen van 1 Tbyte; zie `www.sara.nl/userinfo/teras/description/` voor meer informatie. De Teras wordt beheerd door het nationale rekencentrum SARA in Amsterdam (`www.sara.nl`). Deze supercomputer 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. Gedurende de eerste week van het project wordt een vergadertraining gehouden waarbij per groep de ervaringen met de vergaderingen uit de eerstejaars OGO worden gerecapituleerd, en er openingen worden geboden om de vergaderingen gedurende het tweede jaar nog beter te laten verlopen.

Begeleiding vindt plaats door studenttutoren, die vooral letten op het goed verlopen van het groepsproces. De tutoren zelf hebben geen bijzondere expertise op het probleemgebied of op het gebied van 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 projectcoördinatoren.

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.

1 Beschrijving van de invoer

1.1 Het spoornetwerk

De data over het spoornetwerk bestaat uit twee onderdelen: informatie over de stations, en informatie over de spoorverbindingen. Alle invoerfiles zullen tzt. op de Teras in de directory

```
\home\jfg\invoer
```

staan.

De stations. De informatie over de stations is the vinden in de file `StationsInfo`. Deze file bevat een regel voor elk station, waar achtereenvolgens zijn opgenomen:

- de code van het station: een string bestaand uit hoofd- en kleine letters van ten hoogste 10 karakters.
- de naam van het station: een string van karakters (dit zijn niet alleen hoofd- en kleine letters maar soms ook spaties, haakjes, e.d.), voorafgegaan en afgesloten door dubbele aanhalingstekens (”). De lengte van een stationsnaam is nooit groter dan 60 karakters.
- het type van een station: een integer met de waarde 0 (station tbv. de reguliere dienstregeling), 1 (station dat alleen gebruikt wordt voor bijzondere evenementen) en 2 (een punt in het spoorwegnet zonder station bv. een splitsing of een goederenhalte)
- voor reguliere stations staat er nog de minimale overstaptijd: een integer die de overstaptijd in minuten aangeeft
- verder staat er voor reguliere stations de minimale stoptijd: een integer die de minimale tijd aangeeft die een trein die op het betreffende station stopt minimaal moet stilstaan (om de reizigers te laten in- en uitstappen)

De vijf items zijn gescheiden door een of meer spaties. Voorbeeld:

```
Asra    "Amsterdam Riekerpolder Aansluiting"  2
Aswplz  "Amsterdam Zaanstraat"                2
Atn     "Aalten"                              0 2  1
Az      "Aansluiting Zuid"                    2
Bbg     "Beekbergen"                          1
```

De file `StationsInfo` bevat ten hoogste 1000 regels.

De verbindingen. Het spoornetwerk kan gemodelleerd worden als een (ongerichte) graaf. De knopen in deze graaf zijn de stations, en de edges in de graaf representeren de directe verbindingen tussen de stations. Preciezer: er is een edge tussen stations A en B als er een spoorverbinding tussen A en B is waar geen andere stations aan liggen. Deze informatie is te vinden in de file `VerbindingsInfo`. Voor elke edge staat in deze file een regel, waarin achtereenvolgens zijn opgenomen: de codes van de twee stations die door de edge verbonden worden, de lengte van de verbinding in kilometers, en het aantal sporen van de verbinding (vaak twee, en zeker nooit meer dan tien). Soms is een reizigersverbinding ongeschikt voor reizigersverkeer. Dan staan er 0 sporen aangegeven.

Voorbeeld:

```
Asd  Asdm 3.600 6
Asdm Wgmw 0.573 4
Wgmw Dmn  2.527 2
Dmn  Wgmo 0.943 2
Wgmo Gpda 2.057 2
```

Soms kan het zijn dat twee verbindingen tussen verschillende stations sporen gemeenschappelijk hebben. Dit is gemodelleerd door de punten van type twee in het netwerk.

De stations Amsterdam Sloterdijk en Duivendrecht hebben twee verdiepingen. In de file VerbindingsInfo worden de verkortingen `Assl`, `Assh`, `Dvdl` en `Dvdh` gebruikt om de lage, respectievelijk hoge sporen op deze stations aan te geven. Reizigers kunnen van de hoge naar de lage sporen overstappen.

Elke verbinding staat maar een keer in `VerbindingsInfo`. Dus als de regel “`Ut Uto 2.940 4`” is opgenomen, staat de regel “`Uto Ut 2.940 4`” niet in de file. Overigens zal de invoer grotendeels overeenkomen met het huidige spoornetwerk, maar niet geheel. Het doel is tenslotte om te kijken wat de invloed is van het toevoegen of schrappen van bepaalde verbindingen. Er zijn ten hoogste 1000 verbindingen.

1.2 Het materieel

Om een dienstregeling te kunnen maken is er, naast informatie over het spoornetwerk, ook informatie nodig over het beschikbare materieel en hoe dat ingezet kan worden. Wij gaan uit van het volgende scenario:

- Het aantal treinen dat u in kan zetten is een parameter *aantal_treinen*. U mag zelf bepalen hoe de treinen in de beginsituatie over de stations verdeeld zijn. Als u op een gegeven moment een trein vanuit een bepaald station laat rijden moet die daar op dat moment natuurlijk wel aanwezig zijn. Treinen hoeven 's avonds niet op het vertrekstation van de volgende ochtend te arriveren. Gedurende de nacht is er voldoende tijd de treinen te verplaatsen.

- De maximum snelheid van een trein is m km/uur. Een trein heeft echter enige tijd nodig om op snelheid te komen, en zal aan het eind moeten afremmen. We nemen aan dat een trein a minuten nodig heeft om vanuit stilstand op volle snelheid te komen (of om vanuit volle snelheid tot stilstand te komen), en dat de versnelling (vertraging) lineair is. Daarmee wordt de tijd die een trein minimaal nodig heeft om, vanuit stilstand, een bepaalde afstand af te leggen gegeven door

  $$\text{tijd in minuten} \quad = \quad (\text{afstand in kms})(60/m) + a.$$

  Deze formule geldt alleen als de afstand groot genoeg is, en als de trein aan het eind weer tot stilstand moet komen. Bedenk zelf wat de formule is voor andere gevallen waarbij rekening moet worden gehouden met het feit dat sommige verbindingen zo kort zijn, dat een trein daarin niet op volle snelheid kan komen. Natuurlijk kan u een trein langzamer laten rijden als dat uitkomt, maar ook dan moet er rekening gehouden worden met de optrekvertraging. Een trein die op een verbinding stapvoets moet rijden, kan niet op de volgende spoorsectie zomaar voluit gaan.

- Tenslotte is de minimale afstand tussen twee treinen op hetzelfde spoor een belangrijke parameter. We gaan ervan uit dat er ten minste *min_tijd* minuten tussen twee treinen op hetzelfde stuk spoor moet zitten. Voor het gemak beschouwen we het als afdoende wanneer dit tijdsverschil aanwezig is aan het begin en eindpunt van verbinding, ongeacht de relatieve snelheden van treinen.

Deze parameters staan gespecificeerd in de file `Scenario` in de volgorde: *aantal_treinen*, m, a, *min_tijd*, gescheiden door spaties. Voorbeeld:

```
100 140 4 3
```

betekent dat u 100 treinen ter beschikking hebt, dat de maximum snelheid van een trein 140 km/uur is, dat een trein 4 minuten heeft om op snelheid te komen, en dat er ten minste 3 minuten tussen twee treinen moet zitten.

1.3 De reizigersbewegingen

Om de gemiddelde reistijd te minimaliseren hebt u data nodig over de aantallen reizigers op bepaalde trajecten. Deze informatie is beschikbaar in de file `Verplaatsingen`. Deze file bestaat uit een aantal regels, waarbij elke regel bestaat uit twee codes (het begin- en eindpunt van de reis) en het gemiddeld aantal reizigers tussen de bijbehorende stations. Voorbeeld:

```
Ac    Akm  0
Amr   Aml  5.2
```

Dit geeft aan dat er geen reizigers zijn die van Abcoude naar Akkrum willen, en gemiddeld 5.2 reizigers per dag die van Alkmaar naar Almelo willen. Dit gemiddelde is een indicatie berekend op basis van het aantal verkochte kaartjes, en geeft niet precies het aantal reizigers aan. Wanneer

een bepaalde verplaatsing niet staat opgenomen in de file, mag worden aangenomen dat er een verwaarloosbaar aantal reizigers tussen die plaatsen reizen.

Voor de stations Duivendrecht en Amsterdam Sloterdijk is er vanuitgegaan dat de reizigers van en naar de hoge sporen reizen. De verkortingen `Assl` en `Dvdl` komen niet voor in de file `Verplaatsingen`.

2 De projectopdrachten

Zoals gezegd is het uiteindelijke doel van het project om een dienstregeling te ontwerpen. Preciezer: we willen een zogenaamd basisdagpatroon ontwerpen, d.w.z. de standaard-dienstregeling zoals die op een normale doordeweekse dag van 7:00 tot 23:00 uur wordt uitgevoerd. Dit basisdagpatroon kan beschreven worden in een file die voor elke trein een regel bevat met de volgende informatie:

```
treinnr punt1 vertrek1 punt2 aankomst2 snelheid2 vertrek2 ... puntk aankomstk
```

`treinnr` geeft het nummer van de trein; dit is dus een nummer tussen 1 en *aantal_treinen*. De punten zijn verkortingen van stationsnamen en hulppunten in de topology. De rest van de regel bevat *alle* punten waar de trein doorheen komt, met de aankomsttijd (behalve voor het vertrekpunt) in seconden na 7:00, de vertrektijd (behalve voor het eindpunt) in seconden na 7:00 en de passeersnelheid in m/s als geheel getal. Als een trein niet stopt op een station wordt dat station wel opgenomen in de lijst, waarbij dan aankomsttijd en vertrektijd gelijk zijn. We noemen de op een regel beschreven treinbeweging een *rit*.

2.1 Opdracht 1: De kwaliteit van een basisdagpatroon

Bij opdracht 1 moet u een gegeven basispatroon evalueren. De invoer hiervoor bestaat uit alle hierboven beschreven files, inclusief een file `BasisDagPatroon` die een basisdagpatroon beschrijft in het hierboven gegeven formaat. De opdracht bestaat uit twee onderdelen:

- Ten eerste moet u kunnen bepalen of het gegeven basisdagpatroon toegestaan is. Dit betekent dat aan de volgende voorwaarden voldaan moet zijn:
  - Een rit begint en eindigt op een station waar reizigers in en uit kunnen stappen.
  - Ieder treinnummer komt ten hoogste eenmaal voor in het BasisDagPatroon.
  - Tussen opeenvolgende punten in een rit is een directe verbinding.
  - Als een trein op een bepaald station stopt, dan is de tijd die de trein daar stilstaat groter of gelijk aan de in `StationsInfo` opgegeven minimale stoptijd voor dat station.
  - De vertrektijd, aankomsttijd en doorkomstsnelheid in een punt in een rit zijn in overeenstemming met de hierboven gegeven rijeigenschappen van treinen.
  - De tijd tussen twee treinen die op hetzelfde spoor in dezelfde richting rijden is aan het begin en eind van een verbinding minimaal *min_tijd* minuten, dwz. treinen passeren ieder punt van de spoorbaan met een tijdsverschil van minimaal *min_tijd*. Treinen mogen nooit tussen begin en eindpunt van een verbinding over hetzelfde spoor tegenelkaar in rijden.
- Ten tweede moet u van een toegestaan basisdagpatroon de kwaliteit kunnen bepalen. Zoals hierboven aangeven komt dat in principe neer op de gemiddelde reistijd, waarbij het gemiddelde wordt genomen over de reizigersbewegingen gegeven in `Bewegingen`. Preciezer: stel dat een reiziger van A naar B wil. Om de reistijd te bepalen gaan we ervan uit dat de reiziger op station A arriveert op een willekeurige minuut t op en tussen de 7:00 en 21:00. Dit is een versimpeling, in de zin dat we geen rekening houden met drukkere en minder drukke perioden gedurende een dag. De reistijd wordt dan gemeten als de minimale tijd die de reiziger nodig heeft om, startend op tijdstip t in station A, in station B te komen.

Hierbij wordt er voor elke overstap een extra tijdstraf van 5 minuten bij de reistijd opgeteld. Een overstap wordt natuurlijk alleen gerealiseerd als het verschil in aankomsttijd van de ene en vertrektijd van de andere trein ten minste gelijk is aan de in `StationsInfo` opgegeven minimale overstaptijd voor het betreffende station. Als een reiziger niet van een station naar een ander kan reizen, dan moet een reisduur van 24 uur worden aangehouden.

Uw programma moet dus alle bovenstaande files inlezen. en daarna een uitvoer-file `Kwaliteit` genereren die bestaat uit de string "`incorrect basisdagpatroon`", of wel de string "`correct basisdagpatroon, gemiddelde reistijd is`" gevolgd door de gemiddelde reistijd in seconden.

Deze opdracht is bedoeld om ervaring op te doen met de Teras. Het is slechts nodig dat de opdracht succesvol wordt uitgevoerd en het resultaat aan de coordinatoren wordt doorgegeven. Deze opdracht hoeft niet met een verslag te worden gedocumenteerd en wordt ook niet beoordeeld.

2.2 Opdracht 2: Het genereren van het basisdagpatroon

Voor de tweede opdracht moet u een programma schrijven dat alle bovenstaande files, behalve het basisdagpatroon, inleest en daarna een basisdagpatroon berekent met een zo laag mogelijke gemiddelde reistijd. De uitvoer moet weggeschreven worden in een file `BasisDagPatroon` die het basisdagpatroon beschrijft in het hierboven gegeven formaat.

Voor deze opdracht mag ten hoogste een uur rekentijd worden gebruikt, dwz. de tijd tussen het starten en het termineren van het programma mag ten hoogste een uur bedragen, aannemende dat gedurende dit uur volledige capaciteit van de processoren beschikbaar is.

3 Aanwijzingen en achtergrondinformatie

3.1 Algoritmes en probleemanalyse

Bij deze opdracht gaat het om het ontwerpen van een programma dat snel en correct werkt. Hiertoe is het van belang om eerst een goede analyse van het probleem te maken, en de structuur daarvan goed te doorzien. In deze fase is het van belang om eigenschappen van het probleem die u denkt te gaan gebruiken netjes formuleren en te analyseren.

Daarna zult u de gevonden eigenschappen en inzichten moeten gebruiken om tot een efficient en correct programma te komen. Maak hiertoe eerst een net ontwerp van uw software. Daarnaast is het raadzaam om eerst een eenvoudige, misschien wat minder efficiënte, versie te implementeren; later kunnen dan modules uit die versie vervangen worden door andere, efficiëntere modules met dezelfde functionaliteit.

Uiteraard is het ook noodzakelijk om uw programma uitgebreid te testen. Hierbij kunnen niet alleen bugs in de implementatie aan het licht komen, maar aan de hand van de uitvoer kunt u wellicht ook nieuwe ideeën krijgen om uw programma verder te verbeteren zodat de kwaliteit van de uitvoer omhoog gaat.

Aarzel niet om gebruik te maken van algoritmen uit de literatuur. Het is overbodig te melden dat ook het internet veel informatie geeft, maar die moet wel altijd op zijn merites beoordeeld worden. Vermeld altijd precies de gebruikte bronnen in uw verslag. Probeer ook zoveel mogelijk de tijd en de ruimte die uw algoritmen gebruiken te analyseren, om zodoende inzicht te krijgen in mogelijke bottlenecks in de performance.

3.2 De Teras

Gegevens over de Teras zijn te vinden via `www.sara.nl/userinfo/teras/description/`. Een korte presentatie over de Teras is voorzien in de tweede week. De Teras 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.

3.3 Het IRIX operating systeem

De Teras 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 Teras)
`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 Teras voornamelijk C en C++ ondersteunt, 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 assignment 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. [2]). 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 Teras 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 Teras

Het programma moet op de Teras in batchmode worden uitgevoerd. Op deze manier kan het beste gebruik worden gemaakt van de resources van de Teras. 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 Teras 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.

Zie `www.sara.nl/userinfo/teras/usage/batch/index.html` voor informatie over het opstarten van een batchjob.

Een veel voorkomende fout is om een batchfile op een PC te maken. Het end-of-line character op de PC en de Teras 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 Teras blijven staan, opdat controle mogelijk is, en 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 Teras 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.

Start project, voordracht van NS	2 september
Voordrachten over Teras en bibliotheek	9 september
Vergadertrainingen	ca. derde week
Uitvoeren van opdracht 1 op de Teras	2 oktober
Wederzijdse tussenbeoordeling	16 oktober
Inleveren conceptverslag	30 oktober
Uitslag toegang tot Teras	4 november
Uitvoeren opdracht 2 op Teras	6 november
Presentatie en groepsvoordracht	10 november
Wederzijdse eindbeoordeling, eindbeoordeling en nabespreking	13 november

Tabel 1: Belangrijke data

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.

Projectcoördinator De projectcoördinatoren kunnen 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 T.H.M. (Thijs) Janssen, M.A. (Maurice) Termeer, J.H.J. (Judith) Kennes, S.J.J. (Bas) Kloet, P.J.A. (Paul) van Tilburg en H.A.M. (Harald) Maassen.

   Alle zijn te bereiken via
   (Voorletter.)*.Naam@student.tue.nl.

2. De projectcoördinatoren zijn J.F. Groote (J.F.Groote@tue.nl, HG6.75, tel. 5003/5010) en M. de Berg (m.t.d.berg@tue.nl, HG 6.72, tel 2150).

3. Paul van Tilburg is een studentassistent die helpt bij problemen inzake de te gebruiken programmeertaal en het gebruik van de Teras. Hij kan benaderd worden met implementatievragen. Er zal een wekelijks spreekuur zijn waarvan tijd en plaats op de webpagina bekend gemaakt zal worden (`www.win.tue.nl/~jfg/educ/2I040/2003.html`).

8 Producten en documenten

8.1 Werkplan

In de eerste week moet een werkplan met tijdsplanning worden opgesteld, dat 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 Teras moet bij de projectcoördinatoren 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 Teras vermeld staan, en waarin aanwijzingen gemaakt op het conceptverslag zijn verwerkt. Het eindverslag wordt op papier ingeleverd bij de coördinatoren. Het programma en de output moeten electronisch worden ingeleverd, en moeten beschikbaar zijn op de Teras.

8.5 Presentaties

Er zal een eindpresentatie van 15 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 projectcoördinatoren in overleg met de tutor op grond van het eindverslag en de kwaliteit van het programma. De beoordeling bestaat uit twee onderdelen. Voor beide onderdelen kunt u maximaal 10 punten halen. Het eindcijfer wordt het gemiddelde van de twee deelcijfers. De twee onderdelen zijn (i) het eindverslag, en (ii) de kwaliteit van het door u geproduceerde basisdagpatroon. De groep met het beste basisdagpatroon (d.w.z. de laagste gemiddelde reistijd) krijgt automatisch een 10 voor onderdeel (ii). Groepen die geen geldig basisdagpatroon opleveren krijgen een onvoldoende voor onderdeel (ii).

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), ± (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 op basis van deze proefbeoordeling kan door de projectcoördinatoren wel 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.

Dankbetuigingen. De projectcoördinatoren willen Cor van Woudt en Leo Kroon van NS Reizigers voor hun algemene medewerking. Dank gaat tevens uit naar Twan Laan voor het toeleveren van infrastructuurdata.

Referenties

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

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