Algoritmen en Complexiteit

Algoritmen en Complexiteit

Richting	Informatica,Wiskunde
Jaar	2BINF,Bachelor Wiskunde Keuzevakken

Bespreking

Sinds 2018-2019 is het examen niet meer openboek (door een andere prof), maar de lessen blijven wel een meerwaarde hebben. Er is geen cursus van dit vak, maar de slides zijn duidelijk en alles dat je niet goed begrijpt kan je nalezen in het Sipser boek. Dit vak wordt gegeven door Floris Geerts met Stephen Pauwels als assistent. Het is belangrijk dat je alles goed begrijpt om de oefeningen op te kunnen lossen.

Puntenverdeling

Project reducties: 2/20, Theorie: 18/20.

Examenvragen

Academiejaar 2019-2020- 2de zittijd

Vraag 1: PTIME en NP (5pt)

1. (1.5pt) Hoe wordt de $\theta(t(n))\theta(t(n))$ -tijdscomplexiteit van een niet-deterministische Turing machine gedefinieerd? Hoe verschilt deze definitie van een definitie van de $\theta(t(n))\theta(t(n))$-tijdscomplexiteit van een deterministische Turing machine? Uit je antwoord moet duidelijk zijn:
    1. Wat het belang is van het gebruik van big-oh $\theta(\cdot)\theta(\cdot)$ in de definitie; en
    2. Aan welke voorwaarde Turing machines moeten voldoen zodat deze definitie goed gedifinieerd is.
2. (1.5pt) Geef de definitie van de complexiteitsklasse NP, de definitie van polynomiale tijd reduceerbaarheid, en leg uit wat het wil zeggen dat een taal NP-compleet is. Geef ook een voorbeeld van een NP-complete taal.
3. (2pt) Beschrijf de taal 2SAT en toon aan dat deze taal in PTIME zit. Uit je antwoord moet duidelijk zijn:
    1. Hoe het algoritme voor 2SAT werkt; en
    2. Waarom dat algoritme correct is. Het correctheidsbewijs dat we gezien hebben bestaat uit twee richtingen, je mag hier kiezen welke van de twee richtingen je aantoont. Schrijf wel duidelijk op welke richting je aantoont.

Vraag 2: CLIQUE (5pt)

1. (1pt) Beschrijf het CLIQUE probleem en toon aan dat CLIQUE in NP zit door middel van een polynomiale verifiër.
2. (1.5pt) Beschrijf kort de reductie van 3SAT p p CLIQUE en illustreer dit voor.

```
\(\phi = (x_1 ... x_2 ... x_4) ... NOG VERDER DOEN     \)
```

1. (1.5pt) Toon de juistheid van de reductie 3SAT p p CLIQUE aan. Het juistheidsbewijs van deze reductie bestaat uit twee richtingen. Voor deze vraag mag je kiezen welk van de twee richtingen van het juistheidsbewijs je beschrijft. Schrijf wel duidelijk op welke richting je aantoont.
2. (1pt) Geef een voorbeeld van een probleem waarvoor een reductie vanuit CLIQUE bestaat. Met andere woorden, geef een voorbeeld van een probleem X zodat p p X waar is. Wat wil dit zeggen voor dit probleem X? Welke eigenschap van reducties gebruik je hier? (Vanzelfsprekend verwacht ik een X verschillend van CLIQUE...).

Vraag 3: PSPACE en NPSPACE (5pt)

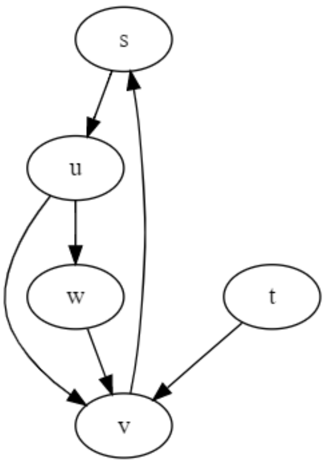
1. (1pt) Formuleer de Stelling van Savitch en toon formeel aan wat hieruit volgt voor de klassen PSPACE en NPSPACE.
2. (3pt) In het bewijs van de Stelling van Savitch laten we de procedure IsBereikbaar los op de configuratie graaf van de uitovering van een Turing Machine M op een input woord w. Antwoord de volgende vragen:
    1. Vul de volgende pseudo-code aan:

```
IsBereikbaar(c1, c2, t)
Als t=1, check [vul aan]
Als t > 1 dan [vul aan]
      Voer IsBereikbaar([vul aan])
      Voer IsBereikbaar([vul aan])
      Als beiden aanvaarden, aanvaard.
Als nog niet aanvaard, verwerp.
```

1.
    1. Hoe wordt uiteindelijk IsBereikbaar uitgevoerd in het bewijs van de Stelling van Savitch? Maak duidelijk wat het onderliggende idee is.
    2. Hoeveel plaats heeft IsBereikbaar nodig, wetende dat de input een θ(s(n))θ(s(n)) -plaats niet-deterministische Turing machine is? Leg zo gedetailleerd mogelijk uit.
2. (1pt) Is NP ⊆⊆ PSPACE of PSPACE ⊆⊆ NP? Wat kan je verder zeggen over de tijdscomplexiteit van problemen in PSPACE? Leg uit.

Vraag 4: LOGSPACE en NLOGSPACE (5pt)

1. (1.5pt) Beschrijf waarom A LBA LB en B LCB LC impliceert dat A LCA LC (transitiviteit). Uit je antwoord moet duidelijk zijn dat je weet wat LOGSPACE reducties zijn en waarom transitiviteit van dergelijke reducties niet vanzelfsprekend is.
2. (1.5pt) Beschouw de onderstaande graaf G. Zoals je ziet zijn er vier knopen bereikbaar vanuit s: s, u, v en w. De knoop t is niet bereikbaar. Wetende dat er vier knopen bereikbaar zijn vanuit s, laat een succesvolle "run" zien op deze graaf van het algoritme dat $\overline{\text{PATH}}$PATH‾ beslist in het bewijs van de Stelling van Immerman-Szelepcsényi. Met andere woorden check dat t niet bereikbaar is. Analyseer de plaatscomplexitiet.



1. (1pt) Toon aan dat PATH NLOGSPACE-hard. In je uitleg, beschrijf duidelijk waarom de gebruikte reductie een LOGSPACE-reductie is.
2. (1pt) Wetende dat PATH NLOGSPACE-compleet is, leg uit hoe hieruit volgt dat 2SAT ook NLOGSPACE-compleet is. In je antwoord is het voldoende om uit te leggen op welke manieren we dit hebben aangetoond in de les. Met andere woorden, voor welk probleem hebben we lidmaatschap in NLOGSPACE aangetoond, van welk probleem hebben we een LOGSPACE-reductie gemaakt, en hoe volgt hieruit dat 2SAT ook NLOGSPACE-compleet is?

Academiejaar 2019-2020- 1ste zittijd

Bestand:A&C 1 2019-2020.pdf

Academiejaar 2018-2019 - 2de zittijd

Media:ANC_Augustus_2019.pdf

Academiejaar 2018-2019 - 1ste zittijd

Nummer al je pagina's in de vorm x/t met t het totaal aantal pagina's en x de huidige pagina. Vanzelfsprekend, markeer ook welke vraag je beantwoord en schrijf leesbaar en structureer je antwoorden op een logische en formele manier. Werk goed door (blijf niet te lang hangen op een bepaalde vraag).

Vraag 1: PTIME en NP (60pt)

1. (25pt) Leg uit wat precies de vraag "PTIME = NP?" betekent. Uit je antwoord moet, onder andere, duidelijk zijn wat voor Turing machines gebruikt worden. Kortom, laat zien dat je de onderliggende definities kent.
2. (10pt) We weten dat SAT een NP-compleet probleem is door de Stelling van Cook-Levin. In het bewijs dat SAT NP-hard is, wordt een algoritme beschreven dat een formule van de vorm ϕ=ϕcell∧ϕstart∧ϕmove∧ϕacceptϕ=ϕcell∧ϕstart∧ϕmove∧ϕaccept construeert. Beantwoord de volgende twee vragen:
    1. Wat is de input van het algoritme dat ϕϕ genereert?
    2. Aan welke eigenschap(pen) moet deze formule voldoen zodat een reductie bekomen wordt?
3. (15pt) Volgend op vraag (2), leg uit hoe ϕmoveϕmove gedefinieerd wordt en waarom dit werkt. Je moet de juistheid van deze formule niet formeel bewijzen. Leg dit in je eigen woorden uit.
4. (10pt) Wat zou een gevolg zijn als 3SAT ∈∈ PTIME? Geef je argumentatie op een formele manier.

Vraag 2: HAMPATH (60pt)

1. (15pt) Toon aan dat HAMPATH = {<G,s,t> | er is een pad van s naar t in de gerichte graaf G die elke knoop slechts eenmaal bezoekt.} in NP zit. Doe dit door middel van een polynomiale verifier.

2. (5pt) We weten dat 3SAT ≤P≤P HAMPATH. Wat betekent dit voor HAMPATH? Leg uit.
3. (10pt) Om 3SAT ≤P≤P HAMPATH aan te tonen gebruikte we een constructie die deels weergegeven is in Figuur 1 (zie hieronder). Als de input van deze constructie een formule is van de vorm $\phi=C1\wedge\ldots\wedge Cm\phi=C1\wedge\ldots\wedge Cm$ over veranderlijken $X=x1,...,xnX=x1,...,xn$, beantwoord dan de volgende twee vragen.
    1. Gegeven Figuur 1, wat is n (het aantal veranderlijken), wat is m (het aantal clauses)?
    2. Vervolledig Figuur 1 zodat de volledige reductie duidelijk is. Merk op dat er geen specifieke formule gegeven is. Je zal dus een deel van de reductie met woorden moeten beschrijven.
4. (15pt) Toon aan dat de vervolledige constructie (volgend op je antwoord in (3)) inderdaad aantoont dat 3SAT ≤P≤P HAMPATH. Het juistheidsbewijs van deze constructie bestaat uit twee richtingen. Voor deze vraag mag je kiezen welk van de twee richtingen van het juistheidsbewijs je beschrijft.
5. (15pt) Beschouw de volgende "boog" variant van HAMPATH: EHAMPATH = {<G,s,t> | er is een pad van s naar t in de gerichte graaf G die elke knoop ten minste eenmaal bezoekt en elke boog maximaal eenmaal doorloopt.}. Maak een polynomiale reductie van HAMPATH naar EHAMPATH.

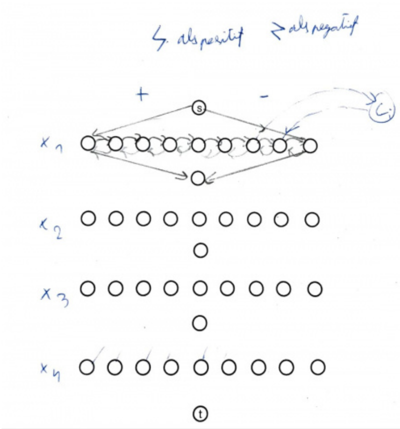


Figure 1: Gedeeltelijke constructie in het bewijs dat 3SAT naar HAMPAH gereduceerd kan worden (Vraag 2.3). Negeer de bijgeschreven dingen (aangezien deze een fout bevatten).

Vraag 3: PATH(90pt)

1. (15pt) Leg uit wat NLOGSPACE betekent. Uit je antwoord moet duidelijk zijn wat voor Turing machines gebruikt worden. Kortom, laat zien dat je de onderliggende definities kent.
2. (20pt) We hebben gezien dat PATH = {<G,s,t> | er is een pad van s naar t in de gerichte graaf G.} een probleem is in NLOGSPACE. Beschrijf het algoritme voor PATH (in pseudocode) dat dit aantoont.
3. (25pt) Is PATH ook NLOGSPACE-compleet? Zoniet, argumenteer waarom. Zoja, beschrijf duidelijk wat je moet aantonen en geef het bewijs.
4. (10pt) Wat heeft de Stelling van Savitch als gevolg voor de deterministische plaatscomplexiteit van PATH? Meer bepaald, volgt hieruit dat PATH ook in LOGSPACE zit?
5. (10pt) Wat zegt de Stelling van Immerman-Szelepcsény over PATH?
6. (10pt) Ken je nog een andere NLOGSPACE-compleet probleem? Wees precies in de beschrijving van een dergelijk probleem (dus niet gewoon de naam van het probleem geven). Leg uit waarom je probleem in NLOGSPACE zit.

Vraag 4: PSPACE (100pt)

1. (15pt) Stel, je definieert PSPACE-compleetheid aan de hand van PSPACE-reducties. Toon aan dat elke niet-triviale taal in PSPACE dan compleet zal zijn. Geef een formeel bewijs. Waarom is dit niet gewenst? Wat is wel een goede definitie van PSPACE-compleetheid?
2. (15pt) We weten dat QBF een PSPACE-compleet probleem is. Leg in woorden het belang van de universele kwantoren uit in het PSPACE-hardheid bewijs. Wat kan je zeggen als er slechts existentiele kwantoren beschikbaar zijn?
3. (15pt) In het bewijs van de Stelling van Savitch laten we de procedure IsBereikbaar los op de configuratie graaf van de uitvoering van een Turing Machine M op een input woord w. Vul de volgende pseudo-code verder aan:

```
IsBereikbaar(c1, c2, t)
Als t=1, check [vul aan]
Als t > 1 dan [vul aan]
      Voer IsBereikbaar([vul aan])
      Voer IsBereikbaar([vul aan])
      Als beiden aanvaarden, aanvaard.
Als nog niet aanvaard, verwerp.
```

1. (15pt) Volgend op vraag (3), hoe wordt uiteindelijk IsBereikbaar uitgevoerd in het bewijs van de Stelling van Savitch? Maak duidelijk wat het onderliggende idee is.
2. (20pt) Hoeveel plaats heeft IsBereikbaar nodig, wetende dat M O(s(n)) plaats nodig heeft. Leg uit.
3. (10pt) Is NP ⊆⊆ PSPACE of PSPACE ⊆⊆ NP? Leg uit.
4. (10pt) Wat kan je zeggen over de tijdscomplexiteit van problemen in PSPACE? Leg uit.

Academiejaar 2017 - 2018 - 1ste zittijd

Media:AC2018ZIT1.pdf

Academiejaar 2015 - 2016 - 2de zittijd

1. Stel f:N→N:n↦n2f:N→N:n↦n2
   - Welke van volgende twee voorwaarde is het strengst voor eender welke functie g:N→Ng:N→N?
     g∈o(f)ofg∈O(f)

     $$g \in o(f) of g \in O(f)$$

     .
   - Geef een voorbeeld van zo een functie gg die aan de zwakkere voorwaarde voldoet, maar niet aan de strenge.
2. Geef een beslissingsboom voor volgende sorteringsalgoritmen voor het sorteren van drie elementen a,ba,b en cc. (tweede algoritme komt nog)

   ifelse(a≤b) then if(b≤c) then return(a,b,c) else if(a≤c) then return(a,c,b)else return(c,a,b)if(c≤b) then return(c,b,a)else if(c≤a) then re
   return(a,b,c) else if(a≤c) then return(a,c,b)else return(c,a,b)elseif(c≤b) then return(c,b,a)else if(c≤a) then return(b,c,a)else
   return(b,a,c)
3. In de cursus staat een constructie om vanuit de vertex cover van een graaf GG in polynomiale tijd een graaf CGCG te maken met een Hamilton circuit. Geef nu een constructie hoe je van deze graaf CGCG een vertex cover van de graaf GG kan vinden. Leg uit waarom je constructie altijd werkt.
4. Als we bij het lemma van Savitch de zesde regel van test-procedure vervangen door
   return(test(C1,C3,k−5)∧(C3,C2,5))

   $$return(test(C1,C3,k-5) \wedge (C3,C2,5))$$

   Wat gaat er dan mis in de redenering?
5. Beschouw het randomized turing machine dat in de cursus geïntroduceerd wordt. Voeg bij de transitietabel volgende transitie toe
   δ(q1,(1,0))=(q3,(1,0),(R,R))

   $$\delta(q1,(1,0))=(q3,(1,0),(R,R))$$

   Bereken dan de kans dat een string van de vorm 0a10b0a10b (i.e. strings van de vorm 0...0010...000...0010...00) geaccepteerd wordt.

Academiejaar 2015 - 2016 - 1ste zittijd

1. Laat in deze vraag f en g functies zijn van ℕ→ℕ
   1. Impliceert f ∈ Θ(g) dat er een polynomiaal verband is tussen f en g? Bewijs!
   2. Geef f en g zodat f ∈ O(g) en zo dat er geen polynomiaal verband is tussen f en g.
2. Pagina 41 (expected tijd voor Select(k, S)).

   Waar komt de term (1-p)T(n) vandaan?

3. Pagina 89 (simulatie van een RAM door een DMTT).

   Wat gaat er in deze redenering precies mis als we het uniform kost criterium zouden gebruiken?

4. Het subgraph isomorphism (SGI) probleem is het volgende. Gegeven 2 grafen G_1 en G_2, beslis of G_2 een subgraaf heeft die isomorf is met G_1. Werk met ongerichte grafen: een graaf is een paar (V, E) waar V een eindige verzameling van kopen is en de verzameling van pijlen E ⊆ V x V is zo dat (x, y) ∈ E ⇔ (y, x) ∈ E. De graaf (V_1, E_1) is een subgraaf van (V_2, E_2) als $V_1 \subseteq V_2$ en $E_1 \subseteq E_2$.

   Is SGI een NP-compleet probleem? Bewijs.

5. Pagina 106 (bewijs van de stelling van Cook)

   Als $E_{5,j,k}$ alleen elementen (i, j, k, i', j', k') bevat met i' ≤ 5, wat kan je dan zeggen over het gedrag van de turingmachine M?

6. Pagina 147 (Synchroon HS algoritme)
   1. Schets een verdeling van de UID's die leidt tot een minimaal aantal boodschappen. (Becshouw om te beginnen een faze 1 voor een proces dat die faze bereikt, en het gedeelte van de ring dat daarbij betrokken is.)
   2. Heeft de verdeling van de UID's invloed op de tijdscomplexiteit? Verklaar.

Academiejaar 2010 - 2011 - 1ste zittijd, apart examen (wegens samenvallende vakken)

1. beschouw volgende programma:

   > read 1, read 3, load =0, store 2, label start, load 3, jzero stop, load 2, mult 1, sub =1, store 2, load 3, sub =1, store 3, jump start, label stop, load 2, sub =2, store 2, write 2, halt

   1. Als we de waarde in register 1 aa noemen en de waarde in register 3 bb, wat berekent dit programma dan?

   2. bereken volgens het logaritmische kostcriterium de plaatscomplexiteit (big oh plus de afleiding).

   3. bereken volgens het uniforme kostcriterium de tijdscomplexiteit (big oh plus de afleiding).

   4. bereken volgens het logaritmische kostcriterium de tijdscomplexiteit (big oh plus de afleiding).

2. Bewerk het algoritme op pagina 45 zo zodat er gebruik wordt gemaakt van deelrijen met grootte 7. Toon aan dat de worst case complexiteit O(n)O(n) blijft.
3. Pas het algoritme op pagina 85 e.v. aan zodat het alle occurences vindt en de complexiteit O(l+n)O(l+n) blijft. Verklaar je antwoord.
4.
   1. Bewijs de eerste stelling op pagina 97 (transitiviteit)
   2. Waar in het bewijs op pagina 109 gebruiken we dat we vertrekken van 3-CNF en niet gewoon van CNF?

Academiejaar 2008 - 2009 - 1ste zittijd

1. Beschouw het RAM-programma:.

   > load =0 store 2 read 1 read 3 load 3 add =1 store 3 label while load 3 jzero endwhile load 2 mult 1 add =2 store 2 load 3 sub =1 store 3 jump while label endwhile write 2 halt

   1. Als we de waarde in register 1 xx noemen en die in register 3 nn, wat berekent dit programma dan?

   2. Wat is volgens het logaritmisch kostcriterium de plaatscomplexiteit van dit programma? Geef de Big-Ohwaarde en de afleiding die hiertoe leidt.

   3. Wat is volgens het uniform kostcriterium de tijdscomplexiteit van dit programma? Geef de Big-Ohwaarde en de afleiding die hiertoe leidt.

   4. Wat is volgens het logaritmisch kostcriterium de tijdscomplexiteit van dit programma? Geef de Big-Ohwaarde en de afleiding die hiertoe leidt.

2. Expected-timecomplexiteit van comparison-based sorteren:

   1. (pagina 57 e.v.) "Neem aan dat alle permutaties van de nn invoerelementen even waarschijnlijk zijn". Waar precies in het bewijs wordt deze veronderstelling gebruikt?

   2. (pagina 58) Leg volgende formule uit

      D(T)=i+D(Tl)+(m−i)+D(Tr)

      $$D(T)=i+D(T_l)+(m-i)+D(T_r)$$

3. (pagina 86) Geef de falingsfunctie van het woordt abababbababab b (over {a,b}{a,b}).

4. (pagina 123)

   1. Wat betekent, in de definitie van AA, de laatste lijn

      ∑i=0p(n)−1(Δ+#)∗#Δi#(Δ+#)∗

      $$\sum_{i=0}^{p(n)-1}(\Delta+\#)^*\#\Delta_i\#(\Delta+\#)^*$$

   2. Welke waarde heeft f(c1,c2,c3)f(c1,c2,c3) als c1,c3∈T∪{#}c1,c3∈T∪{#}, c2=[q,a]c2=[q,a] en δ(q,a)=(p,d,R)δ(q,a)=(p,d,R)?

5. (Min-Cut) Hoe groot moet rr gekozen worden om ervoor te zorgen dat er tenminste 99% kans het volgende geldt?

   > De gevonden cut heeft niet meer dan 10% meer edges dan een min-cut

Academiejaar 2006 - 2007 - 1ste zittijd

1. Geef de big-oh tijdscomplexiteit van:

   1. > function nestedloops(int m, n) { int i=1; while (i <= m) { int j = 1; while (j <= n) { j = j+1; } i = i + 1; } }

   2. > function nestedloops(int m, n) { int i=1; int j = 1; while (i <= m) { while (j <= n) { j = j+1; } i = i + 1; } }

2. Pas maximum 2 regels aan in 1.(b) om een complexiteit te bekomen van O(m * log(n)).

3.
    1. Waarom is op pagina 63 het selecteren van het k'de element minder complex dan de quicksort?
    2. Vorm STORE * 30 om naar een opeenvolging van RASP-instructies.
    3. Context: p. 66. Deze redenering geldt enkel voor sets van invoerelementen die allemaal verschillend zijn. Waarom volstaat dat?
4. p. 85 en volgende:
    1. Veralgemeen het algoritme zodat er in de occurrence één fout toegelaten wordt.
       input: woorden x, y; met x=a1…anx=a1…an, y=b1…bly=b1…bl.
       output: het eerste deelwoord ai+1…ai+lai+1…ai+l in ten hoogste 1 letter verschilt van y.
       Hint: gebruik toestanden q(u,v)q(u,v) met uu en vv woorden van lengte ≤l≤l die in ten hoogste 1 positie verschillen. Geef aan welke transities met dergelijke q(u,v)q(u,v) verbonden zijn.
    2. Heeft dit algoritme een wordt-case tijdscomplexiteit in O(n)?
5. Context: p. 134, klasse RP. Bewijs dat het vervangen van de 1212 door 1313 in de definitie niets verandert; m.a.w. dat op die manier nog steeds dezelfde klasse gedefinieerd wordt.

Academiejaar 2006 - 2007 - 2de zittijd

1. Gegeven volgend RAM programma:

   > read 0 store 1 sub =1 store 5 load =6 store 2 label loop load 1 jzero end sub =1 store 1 read *2 write *2 load 2 add =1 store 2 jump loop label end write = -1 read 3 load =6 add 5 load *0 store 4 label for_init load 5 label for_inc sub =1 store 2 jgtz for_body jzero for_body jump end_for label for_body load 3 mult 4 store 4 load =6 add 2 load *0 add 4 store 4 load 2 jump for_inc label end_for write 4

    1. Wat is de Big Oh tijdscomplexiteit volgens het uniform kost criterium voor volgend programma? + een 3-tal lijnen uitleg (vertrek hierbij zonder rekening te houden met Big Oh en zet dan om)
    2. Wat is de Big Oh plaatscomplexiteit volgens het logaritmisch kost criterium voor dit programma? + een 3-tal lijnen uitleg
2. Teken de beslissingsbomen van volgende sorteeralgoritmen:

   > Algoritme 1: if (a < b) then if (b < c) then return (a,b,c) else if (a < c) then return (a,c,b) else return (c,a,b) else if (c < b) then return (c,b,a) else if (c < a) then return (b,c,a) else return (b,a,c)
   >
   > Algoritme 2: if (a < b) then if (b < c) then return (a,b,c) else if (a < c) then return (a,c,b) else return (c,a,b) else x := a; a := b; b := x; if (c < a) then return (c,a,b) else if (c < b) then return (a,c,b) else return (a,b,c)

3. Bewijs de eerste stelling van pagina 97 (wees precies!)
4. Laat HP het volgende beslissingsprobleem zijn:
   Gegeven: een gerichte graaf G.
   Gevraagd: heeft G een Hamiltonian Path, i.e. een gericht pas waarop elke knoop precies éénmaal voorkomt?
   Geef een polynomiale reductie van HC (Hamiltonian Circuit) naar HP.

Academiejaar 2005 - 2006 - 1ste zittijd

1. Beschouw het RAM programma dat weergegeven wordt in fig. 1.
    1. Wat is volgens het uniforme kost criterium de plaatscomplexiteit van dit programma? Geef de Big-O waarde en de afleiding die hiertoe leidt.
    2. Wat is volgens het uniforme kost criterium de tijdscomplexiteit van dit programma? Geef de Big-O waarde en de afleiding die hiertoe leidt.
    3. Wat is volgens het logaritmische kost criterium de Big-O plaatscomplexiteit van dit programma?
2. Stel dat we een beperkte soort RAM invoeren die geen MULT instructie heeft.
    1. Kan de beperkte RAM de gewone RAM simuleren? M.a.w. bestaat er voor elk gewoon RAM programma een equivalent programma voor de beperkte RAM?
    2. Is het zo dat de tegenhanger van de stelling op p.21 geldt, m.a.w. dat er een constante k bestaat zodat de logaritmische tijdscomplexiteit slechts met ten hoogste een factor k toeneemt bij simulatie van een RAM programma door de beperkte RAM?
3. Zie p.43, 2de bullet:
    1. Geef aan hoe NonEmpty[l]NonEmpty[l] eenvoudig geïnitialiseerd kan worden, gegeven de lexografisch geordende koppels (j,sij)(j,sij).
    2. Hoe kom je aan die O(m+ltot)O(m+ltot)?
4. Geef de falingsfunctie en de automaat MyMy voor y=ababby=ababb.