Tentamen combinatorische optimalisatie 26-05-2014. Tijd: 9.00-11.30

Tentamen is met 'gesloten boek'. Beschrijf bij elke opgave steeds het belangrijkste idee. Notatie en exacte formulering is van minder belang. Een tekening kan soms handig zijn. Ook voor een half antwoord kun je punten krijgen.

De vragen 7 en 8 zijn de toetsvervangende vragen. Maak deze alleen als je de tussentoes niet hebt gemaakt of als je de tussentoets wilt vervangen door deze vragen.

Opgave	1	2	3	4	5	6*a*	6*b*	7	8
Max pntn	5	5	10	10	10	5	5	5	5

Opgave 1 (5punten).

Beantwoord de volgende vragen alleen met ja (waar), nee (niet waar) of laat open. Goed antwoord: $+1$ punt; Fout antwoord: -1 punt; Geen antwoord: +0 punt. (Minimaal aantal punten is 0 punten.)

(a) Stel Π is een *NP*-compleet beslissingsprobleem. Als $\Pi \in P$ dan geldt $P = NP$.

(b) Stel Π_1 en Π_2 zijn beslissingsproblemen in *NP*. Als Π_2 *NP*-compleet is en Π_1 reduceert tot Π_2 dan is Π_1 ook *NP*-compleet.

(c) Het (integer) knapsack probleem is *sterk* NP-moeilijk (strongly NP-hard).

(d) Het minimum spanning tree probleem (MST) is reduceerbaar tot het traveling salesman probleem (TSP).

(e) Het TSP probleem waarbij alle afstanden 1 of 2 zijn is *NP*-moeilijk (NP-hard).

Opgave 2 (5 punten)

Beschouw onderstaande instantie van het probleem $1|r_j, pmtn|\sum_j C_j$. (Dwz, er is 1 machine, job j heeft release date r_j, preemption is toegestaan, en het doel is het minimaliseren van de som van completeringstijden.)
Geef een optimaal schedule en z'n waarde voor deze instantie en beschrijf ook het optimale algoritme.

jobs	1	2	3	4
r_j	0	0	1	2
p_j	6	4	2	2

Opgave 3 (10 punten)

Stel $G = (V,E)$ is een simpele graaf. (Een graaf is simpel als het geen multiple edges of loops bevat.) Verder is G ongericht en verbonden en heeft elke lijn $e \in E$ een gewicht $w(e) \geq 0$. De lijnen van G zijn gelabeld zodat $w(e_1) \leq w(e_2) \leq \cdots \leq w(e_m)$. Stel $w(e_{m-1}) < w(e_m)$ (stricte ongelijkheid) en neem verder aan dat G

verbonden blijft als e_m weggelaten wordt. Laat zien dat *geen enkele* Minimum Spanning Tree van G lijn e_m bevat.

Opgave 4 (10 punten) Gegeven zijn n soorten muntjes met waarden $x_1, x_2, \ldots, x_n$ en van elke soort is er een onbegrensde voorraad. Met deze muntjes willen we een bedrag W exact betalen, met andere woorden, we willen een aantal muntjes pakken welke samen precies waarde W hebben. Dit is niet altijd mogelijk. Bijvoorbeeld, als we alleen muntjes van waarde 5 en 10 hebben dan kunnen we wel 25 betalen maar niet 12. Geef een $O(nW)$ dynamic-programming algoritme om gegeven de waarde W te bepalen of het mogelijk is om bedrag W exact te betalen.

Hint: Voor alle $v \in \{0, 1, \ldots, W\}$ definieer $F(v) = 1$ als het mogelijk is om een bedrag v te betalen en laat $F(v) = 0$ als dat niet mogelijk is.

(Je hoeft geen algoritme uit te schrijven maar beschrijf alleen de essentie. Welke waarden staan en er in de dynamic programming tabel? Hoe worden deze berekend? Waarom kost dit alles $O(nW)$ tijd?)

Opgave 5 (10 punten) Een instantie van het *minimum Steiner Tree* probleem wordt gegeven door een volledige graaf $G = (V, E)$ en verzameling punten $V' \subseteq V$ en afstanden d_{uv} tussen elk paar punten $u, v \in V$. Een oplossing is een boom T welke alle punten van V' (en mogelijk meer punten) bevat. In het voorbeeld hieronder zijn de punten van V' donker gekleurd. Doel is om een oplossing met minimale kosten (=som van de afstanden) te vinden. Stel dat de afstanden aan de driehoeksongelijkheid voldoen (triangle inequality).

Geef een polynomiaal algoritme voor het minimum Steiner Tree probleem waarvoor geldt dat de waarde hooguit twee maal de optimale waarde is. De polynomiale looptijd hoef je niet aan te tonen. Toon wel aan dat de waarde hooguit twee maal de optimale waarde is.

Hint: Vul getallen op de puntjes in en beargumenteer waarom dit geldt.

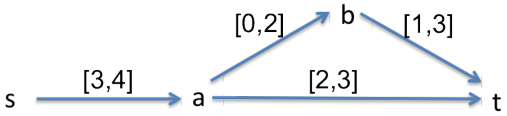
$OPT(MST)$:	*de waarde van de minimal spanning tree.*
$OPT(TSP)$:	*de waarde van de beste TSP oplossing.*
$OPT(ST)$:	*de waarde van de beste Steiner tree.*

$$...\mathrm{OPT}(MST) \leq ...\mathrm{OPT}(TSP) \leq ...\mathrm{OPT}(ST).$$

Opgave 6 Het algoritme van Ford en Fulkerson vindt een maximale s,t-stroom in een gericht netwerk met capaciteiten op de pijlen. Stel nu dat er voor elke lijn a naast de capaciteit $c(a)$ (de bovengrens) ook een ondergrens $b(a)$ gegeven is. Voor elke pijl a moet dus gelden dat de waarde van de stroom ligt in het interval $[b(a),c(a)]$. Dit is echter niet altijd mogelijk voor een s,t-stroom.

(a) (5 punten) Geef een voorbeeld van een gericht netwerk met punten s en t en onder- en bovengrenzen waarbij er geen toegelaten stroom van s naar t bestaat.

(b) (5 punten) Het probleem van het vinden van een toegelaten stroom in een netwerk met onder- en bovengrenzen is te reduceren tot het vinden van een maximale stroom in een netwerk met alleen bovengrenzen. (De standaard vorm waarvoor we Ford-Fulkerson kunnen gebruiken). Laat zien hoe deze reductie werkt door het toe te passen op onderstaande instantie.



Onderstaande vragen 7 en 8 alleen beantwoorden als je de tussentoes niet hebt gemaakt of als je de tussentoets wilt vervangen door onderstaande vragen.

Opgave 7 (5 punten) n muntjes worden onafhankelijk van elkaar opgegooid waarbij de kans dat munt i op de kop-zijde belandt gelijk is aan p_i ($i = 1 \ldots n$). Voor een gegeven $k \leq n$ wil je de kans berekenen dat precies k van de muntjes op de kop-zijde belanden. Geef hiervoor een $O(n^2)$ dynamic programming algoritme. (Je mag aannemen dat vermenigvuldigen en optellen van twee getallen in constante tijd ($O(1)$-time) kan.)

Hint: Nummer de muntjes $1,2,\ldots,n$. *Stel* $P(i,j)$ *is de kans dat met de eerste* i *muntjes samen precies* j *maal kop wordt gegooid. (*$i \in \{1,\ldots,n\}$, $j \in \{0,\ldots,k\}$*).*

(Je hoeft geen algoritme uit te schrijven maar geef alleen de essentie. Geef aan hoe de dynamic programming tabel eruit ziet. Hoe worden de waarden in de tabel uitgerekend? Waarom kost dit alles $O(n^2)$ tijd?)

Opgave 8 (5 punten) Beschrijf minimaal twee verschillende 2-approximatie algoritmes voor het TSP probleem (met driehoeksongelijkheid). De beschrijving is voldoende. Er wordt *niet* gevraagd om de factor 2 te bewijzen.

Antwoorden

1 (a) Ja (b) nee (c) nee (d) ja (e) ja.

Toelichting: (a) Per definitie van *NP*-compleetheid is elk probleem reduceerbaar tot Π. Als $\Pi \in P$ dan is elk probleem in *NP* dus op te lossen in polynomiale tijd door het te reduceren naar Π.

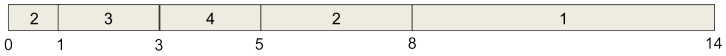
(b) Andersom: Als Π_1 *NP*-compleet is en Π_1 reduceert tot Π_2 dan is Π_2 ook *NP*-compleet.

(c) Nee, knapsack (en integer knapsack) is op te lossen in pseudo polynomiale tijd door middel van dynamic programming. (Zie slides en complexity notes.)

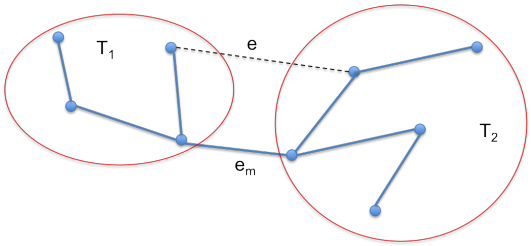
(d) Ja, elk probleem in NP is reduceerbaar tot een NP-moeilijk probleem.

(e) Ja, het Hamiltonian Cycle probleem is reduceerbaar tot het TSP probleem met afstanden 1 en 2. Zie Stelling 1, complexity notes.

2 Het optimale algoritme scheduled altijd in volgorde van Shortest Remaning Processing Time (SRPT). Dat geeft hier onderstaand schedule. De waarde is $C_1 + C_2 + C_3 + C_4 = 14 + 8 + 3 + 5 = 30$.



3 Stel dat we een MST T hebben die e_m wel bevat. We laten zien dat dit niet mogelijk is. Laat e_m weg uit T. De boom valt dan uiteen in twee delen, zeg T_1 en T_2. Omdat G verbonden blijft als e_m wordt wegelaten moet er dus nog een lijn $e \in E$ zijn die T_1 met T_2 verbindt. Voeg nu e toe na het weglaten van e_m. We krijgen dan weer een boom en het totale gewicht is afgenomen met $w(e_m) - w(e) > 0$. Dit is niet mogelijk want we hadden aangenomen dat T een minimale opspannende boom is.



4 Een bedrag v kan exact betaald worden als er een muntje i is zodat het bedrag $v - x_i$ exact betaald kan worden.

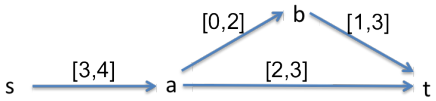
Neem $F(0) = 1$. Er geldt nu dat $F(v) = 1$ als er een $i \in \{1, \ldots, n\}$ is waarvoor $F(v - x_i) = 1$. Om $F(v)$ te berekenen hoeven we dus maar n waarden op te zoeken in de tabel. In de tabel staan alle waarden $F(v)$ voor $v = 0, \ldots, W$ en elke kan in $O(n)$ tijd berekend worden. Het eindantwoord is $F(W)$.

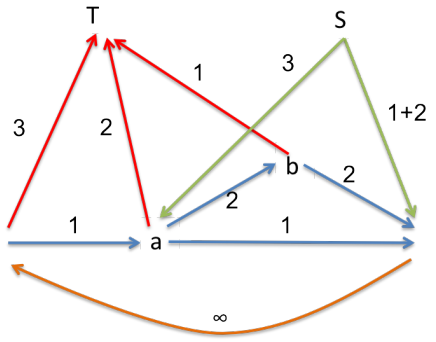
5 Als algoritme nemen we gewoon een minimum spanning tree.

$$\mathrm{OPT}(MST) \leq \mathrm{OPT}(TSP) \leq 2\mathrm{OPT}(ST).$$

Bekijk een optimale TSP en laat een lijn weg. We krijgen dan een opspannende boom: $OPT(MST) \leq \mathrm{OPT}(TSP)$.
Bekijk een optimale Steiner tree en verdubbel alle lijnen in de oplossing. Maak een Eulertour en pas vervolgens shortcuts toe zodat een TSP tour verkregen wordt: $OPT(TSP) \leq 2\mathrm{OPT}(ST)$.

6





De ondergrenzen worden vervangen door 0 en van de bovengrens wordt de ondergrens afgehaald. Van s naar t komt een arc met oneindige capaciteit en er worden nieuwe S en T toegevoegd. Voor elke pijl a met ondergrens $b(a)$ gaat er een nieuwe pijl met capaciteit $b(a)$ naar T en een pijl met capaciteit $b(a)$ vanuit S. Er bestaat een toeglaten stroom in het originele netwerk, dan en slechts dan, als er een stroom van S naar T met waarde 6 bestaat in het nieuwe netwerk.

7 Er geldt

$$P(i,j) = p_i P(i-1,j-1) + (1-p_i)P(i-1,j).$$

In de tabel zetten we de waarden $P(i,j)$ voor alle $i \in \{1,\ldots,n\}$ en $j \in \{0,\ldots,k\}$. Dit zijn $O(n^2)$ waarden en het uitrekenen van een enkele waarde gaat in $O(1)$ tijd. Het antwoord wordt gegeven door $P(n,k)$.

8 Algortme 1: Neem een MST en verdubbel alle lijnen. Dit geeft een Euler tour. Pas vervolgens shortcuts toe om een TSP tour te krijgen.

Algortme 2: Bijvoorbeeld Nearest Addition: Start met een enkel punt en breid de tour T steeds verder uit. Kies steeds een punt k die niet in T zit en waarvoor de afstand tot T minimaal is. Stel j is het punt op T met minimale aftstand en stel i is buur van j op T. Voeg de lijnen (j,k) en (i,k) toe en laat (i,j) weg. Herhaal dit totdat alle punten op de tour liggen.