Operations Research Society of South Africa

Voorlegging vir publikasie in ORiON

Operasionele Navorsings-vereniging van Suid-Afrika

'n Voorbeeld vir die LaTeX-pakket `ORiONafr.sty`*

Identiteite van outeurs onderdruk: Beoordelaarskopie

Opsomming

Die styl van artikels wat in ORiON gepubliseer word, word gegee. Voorbeelde van hoe omgewings soos tabelle ingesluit moet word, word verskaf. Die styl en standaard van ORiON word regdeur hierdie dokument gehandhaaf. Die doel is nie om 'n wiskundig korrekte of samehangende inhoud te verskaf nie, maar 'n dokument aan die hand van standaard LaTeX-voorbeelde saam te stel.

Sleutelwoorde: Stylblad, voorbeelde, LaTeX

Abstract

All papers submitted in Afrikaans must be accompanied by an extended abstract of at least 300 words in English. This Eglish abstract must be a summary of the whole paper. It should give the reader a proper overview of the article.

Key words: Style sheet, examples, LaTeX

1 Inleiding

Hierdie artikel bevat slegs voorbeelde van verskeie funksies vir gebruik met die ORiON-stylblad saam met LaTeX. Die dokument moet dus nie gelees word vir sinvolle inhoud of korrekte logika nie. Die stellings, definisies ens. wat hier verskaf word mag moontlik nie sin maak nie. Die doel is egter nie om korrekte stellings en definisies te verskaf nie, maar om voorbeelde van hierdie entiteite te verskaf. Die voorbeelde toon hoe sulke standaardomgewings en elemente binne die ORiON-stylblad geskep word. Die artikel gee ook voorbeelde van die voorkeurstyl waarin ORiON vergelykings, verwysings en ander standaardelemente in 'n artikel hanteer.

Die voorbeelde wat hier verskaf word, is nie 'n allesomvattende versameling nie. Indien 'n outeur baie spesifieke of uitsonderlike situasies binne 'n artikel wil insluit, kan daar met die besigheidsbestuurder van ORiON geskakel word. Vrae wat gereeld ontstaan, sal as voorbeelde in opdaterings van hierdie dokument opgeneem word.

*Hierdie dokument word deur ONSA befonds

Daar is verskeie handleidings beskikbaar om LaTeX en TeX aan te leer [2, 5]. Indien hierdie handleidings nie voldoende is nie, kan die internet ook geraadpleeg word. Teks word in die brondokument ingevoer soos wat die geval met gewone LaTeX-dokumente is. Die omgewings in Tabel 1 is reeds gedefinieer.

`afleiding`	`aksioma`	`algoritme`
`definisie`	`gevolgtrekking`	`heuristiek`
`lemma`	`proposisie`	`stelling`
`voorbeeld`		

Tabel 1: *Reeds gedefinieerde omgewings binne* `ORiONafr.sty`

2 Teks

Om LP's met die simpleksalgoritme op te los, is kennis van lineêre algebra nodig. Onder andere is die begrip van vektore wat lineêr afhanklik is, belangrik.

Definisie 1 *Indien* $\underline{a}_j$ *met* $j = 1, 2, 3, \ldots, r$ *'n versameling vektore is, dan het die vergelyking*

$$k_1\underline{a}_1 + k_2\underline{a}_2 + \ldots + k_r\underline{a}_r = \underline{0}$$

ten minste een oplossing, naamlik $k_1 = k_2 = \ldots = k_r = 0$. *Indien dit die enigste oplossing is, word die versameling van* $\underline{a}_j$ *lineêr onafhanklik genoem en indien daar ander oplossings bestaan, word die versameling lineêr afhanklik genoem.*

Die versameling kolomvektore, $\underline{a}_j$ met $j = 1, 2, 3, \ldots, k$, is lineêr afhanklik of lineêr onafhanklik. Gestel die versameling vektore is lineêr afhanklik. Onder hierdie omstandighede is dit moontlik om die aantal positiewe veranderlikes stap vir stap te verminder totdat die kolomme wat met die oorblywende positiewe veranderlikes ooreenstem lineêr onafhanklik is. Indien die $\underline{a}_j$, met $j = 1, 2, 3, \ldots, n$, lineêr afhanklik is, bestaan daar reële getalle α_j wat nie almal nul is nie, sodat

$$\sum_{j=1}^{k} \alpha_j\underline{a}_j = \underline{0}. \tag{1}$$

Vergelyking (1) kan gebruik word om 'n x_j, sê x_r, in

$$\sum_{j=1}^{k} x_j\underline{a}_j = \underline{b}, \qquad x_j > 0 \tag{2}$$

te reduseer tot nul. Gestel enige vektor a_r van die k vektore in vergelyking (1) waarvoor $\alpha_r \neq 0$ kan in terme van die oorblywende $k - 1$ vektore uitgedruk word. Dus is

$$\underline{a}_r = -\sum_{j\neq r} \frac{\alpha_j}{\alpha_r}\underline{a}_j. \tag{3}$$

Stel vergelyking (3) nou in vergelyking (2), sodat volg dat

$$\sum_{j=1}^{p}\left(x_j - x_r\frac{\alpha_j}{\alpha_r}\right)\underline{a}_j = \underline{b}. \tag{4}$$

Hierdie oplossing het nie meer as $k-1$ nie-nul veranderlikes nie. Indien $\underline{a}_r$ arbitrêr gekies word, is daar geen versekering dat van die veranderlikes nie dalk negatief kan wees nie. Indien $\underline{a}_r$ egter reg gekies word, sal die oorblywende $k-1$ veranderlikes steeds nie-negatief wees. Kies $\underline{a}_r$ só dat

$$x_j - x_r\frac{\alpha_j}{\alpha_r} \geq 0 \quad j = 1, \ldots, p. \tag{5}$$

Vir enige j waarvoor $\alpha_j = 0$, sal vergelyking (5) outomaties bevredig word. Indien $\alpha_j \neq 0$, lewer vergelyking (5), gedeel deur α_j, een van die volgende ongelykhede

$$\frac{x_j}{\alpha_j} - \frac{x_r}{\alpha_r} \geq 0 \quad \text{indien } \alpha_j > 0, \text{ of} \tag{6}$$

$$\frac{x_j}{\alpha_j} - \frac{x_r}{\alpha_r} \leq 0 \quad \text{indien } \alpha_j < 0. \tag{7}$$

Uit (6) en (7) volg dat $\frac{x_r}{\alpha_r}$ nie groter as enige $\frac{x_j}{\alpha_j} > 0$ kan wees nie en ook nie kleiner as enige $\frac{x_j}{\alpha_j} < 0$ nie. Gestel dus ons kies die vektor $\underline{a}_r$ só dat

$$\frac{x_r}{\alpha_r} = \min_j\left\{\frac{x_j}{\alpha_j}, \quad \alpha_j > 0\right\}. \tag{8}$$

Dan sal elke veranderlike in vergelyking (4) nie-negatief wees. 'n Toelaatbare antwoord met nie meer as $k-1$ nie-nul veranderlikes is gevind. Indien hierdie proses van redusering voortgesit word, sal die oorblywende x_j's uiteindelik lineêr onafhanklik wees en sal die oplossing 'n basiese toelaatbare oplossing wees.

Die stelling wat die teorie oor die oplossing van LP's met behulp van die simpleksalgoritme goed saamvat, is die fundamentele stelling van LP.

Stelling 2 (Fundamentele stelling van lineêre programmering (LP)) *Die stelling bestaan uit twee dele:*

1. *Indien daar 'n toelaatbare oplossing vir 'n LP is, bestaan daar 'n basiese toelaatbare oplossing.*
2. *Indien daar 'n optimale oplossing vir 'n LP is, bestaan daar 'n optimale basiese toelaatbare oplossing.*

Slegs 'n gedeeltelike bewys vir Stelling 2 word hier aangebied.

Bewys: Aanvaar dat daar 'n toelaatbare oplossing bestaan met $k < n$ veranderlikes wat 'n positiewe waarde het. Nommer hierdie veranderlikes só dat die eerste k veranderlikes 'n positiewe waarde het. Dan kan die oplossing geskryf word as

$$\sum_{j=1}^{k} x_j\underline{a}_j = \underline{b}$$

met $x_j > 0$ vir $j = 1, 2, \ldots, k$ en $x_j = 0$ vir $j = k+1, \ldots, n$. Die vektore $\underline{a}_j$ wat met die positiewe waardes geassosieer is, kan òf lineêr onafhanklik òf lineêr afhanklik wees [6].

Vir 'n meer volledige bewys kan [4] geraadpleeg word.

LP's kan natuurlik baie maklik opgelos word met behulp van rekenaarpakkette. 'n Paar pakkette word in Tabel 2 gelys.

Pakket	Voordele
LINDO & LINGO	Programmeerbaar
	Ingeboude wiskundige funksies
QM for Windows	Gebruikersvriendelik
	Koppelvlak met Excel

Tabel 2: *LP-sagteware met hul voordele.*

3 Algoritmes

Die binêre soekalgoritme begin in die middel van 'n alfabetiese lys van woorde soek. Die algoritme vergelyk die middelste woord met die gesoekte woord, WRD, om te toets of die middelste woord WRD is. Indien dit nie die geval is nie beteken dit òf WRD kom voor òf WRD kom ná die middelste woord in die alfabetiese lys. Indien WRD voor die middelste woord kom, word die woord halfpad tussen die eerste en die middelste woord in die lys getoets om te bepaal of WRD gelyk aan daardie woord is, terwyl die tweede helfte van die lys afgesny word. Indien WRD ná die middelste woord kom, word die woord halfpad tussen die middelste en die laaste woord in die lys getoets om te bepaal of WRD gelyk aan daardie woord is, terwyl die eerste helfte van die lys afgesny word. Hierdie prosedure van halvering van die oorblywende lys word herhaaldelik toegepas totdat die woord gevind is of die lys uitgeput is. Dit impliseer dat slegs die helfte van die lys ná die eerste stap en 'n kwart van die lys n'a die tweede stap en 'n agste van die lys ná die derde stap, ensovoorts, deursoek hoef te word. (Met die middelste woord word bedoel die woord wat in die $\left\lfloor \frac{1+n}{2} \right\rfloor$-de posisie staan.)

Algoritme 3 (Binêre soekalgoritme) *Soek 'n woord in 'n alfabetiese lys van woorde deur die middelste woord van die lys met die gesoekte woord te vergelyk en die lys, indien nodig, te verdeel in sublyste.*

1. *Laat α en β die eerste en laaste posisies van 'n sublys wees. Aanvanklik sal $\alpha = 1$ en $\beta = n$. Dus $\alpha \leftarrow 1$ en $\beta \leftarrow n$. WRD $\leftarrow$ "die gesoekte woord."*

2. *Herhaal hierdie stap solank $\alpha \leq \beta$, so nie gaan na 3.*

   (a) *Bepaal k, die middelste posisie (tussen α en β).*

$$k \leftarrow \left\lfloor \frac{\alpha + \beta}{2} \right\rfloor$$

(b) Indien $W(k) = WRD$, stop. Afvoer: k.

(c) As $WRD > W(k)$, dan $\beta \leftarrow k - 1$. Gaan na 2a.

(d) As $WRD < W(k)$, dan $\alpha \leftarrow k + 1$. Gaan na 2a.

3. Stop. Afvoer: "Nie in die lys nie".

Kom ons ondersoek nou die (swakste-gedrag-) kompleksiteit van die binêre soekalgoritme. Let op dat beide stap 1 en stap 3 slegs een keer uitgevoer word. In elkeen van hierdie stappe word 'n konstante hoeveelheid operasies uitgevoer. Dit beteken dat hierdie stappe se kompleksiteit $O(1)$ is. Laat $B(n)$ die maksimum aantal kere wees wat stap 2 herhaal moet word. Nadat stap 2 die eerste keer uitgevoer is, sal die lys wat oorbly 'n maksimum van $\lfloor \frac{n}{2} \rfloor$ elemente oorhê om te oorweeg. Dit beteken

$$B(n) = 1 + B\left(\left\lfloor \frac{n}{2} \right\rfloor\right). \tag{9}$$

Ons gebruik induksie om te bewys dat

$$B(n) \leq 1 + \log_2 n \tag{10}$$

vir elke positiewe heelgetal n. Vergelyking (10) geld indien $n = 1$. Aanvaar hierdie resultaat is waar vir een of ander heelgetal, k. Dan moet ons aantoon dat hierdie ongelykheid ook geld vir $k + 1$. Ons aanvaar dus dat

$$B(k) \leq 1 + \log_2 k. \tag{11}$$

Volgens vergelyking (9) weet ons dat

$$B(k+1) = 1 + B\left(\left\lfloor \frac{k+1}{2} \right\rfloor\right),$$

maar uit vergelyking (11) weet ons dat

$$B\left(\left\lfloor \frac{k+1}{2} \right\rfloor\right) \leq 1 + \log_2 \left(\left\lfloor \frac{k+1}{2} \right\rfloor\right).$$

Verder weet ons dat

$$\log_2 \left(\left\lfloor \frac{k+1}{2} \right\rfloor\right) \leq \log_2 \left(\frac{k+1}{2}\right).$$

Dit beteken dat

$$\begin{aligned} B(k+1) &\leq 2 + \log_2 \left(\frac{k+1}{2}\right) \\ &\leq 2 + \log_2(k+1) - \log_2(2) \\ &\leq 1 + \log_2(k+1), \end{aligned}$$

wat die verlangde resultaat lewer. Ons kan dus aflei dat $B(n) = O(\log_2 n)$.

4 Matrikse

Die veranderlike, x_3, gaan met 'n waarde van $\frac{-20}{-1} = 20$ in die basis inkom, wat kleiner of gelyk aan sy bogrens is. Dus kan die simpleksiterasie uitgevoer word. Die kolom onder x_3 word gegee deur

$$B_1^{-1}p_3 = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix}.$$

Die elementêre matriks word gegee deur

$$E_1 = \begin{bmatrix} \left(\frac{1}{-1}\right) & 0 & 0 \\ -\left(\frac{-1}{-1}\right) & 1 & 0 \\ -\left(\frac{0}{-1}\right) & 0 & 1 \end{bmatrix}.$$

Ons bereken vervolgens

$$B_2^{-1} = E_1 B_1^{-1} = \begin{bmatrix} -1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & -1 \\ 0 & 0 & -1 \end{bmatrix}.$$

Ná die iterasie word die onderskeie groothede gegee deur

$$A' = \begin{bmatrix} -1 & -1 & 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 \\ -1 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \quad B_2^{-1} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & -1 \\ 0 & 0 & -1 \end{bmatrix} \quad \underline{b} = \begin{bmatrix} -60 \\ 30 \\ -80 \end{bmatrix}$$

$$\begin{aligned} \underline{c}_{bv}(2) &= [-1, 0, -1] & \underline{c}_{nbv}(2) &= [-2, 0, 0] \\ \underline{x}_{bv}(2) &= [x_3, s_2, \widetilde{x}_2] & \underline{x}_{nbv}(2) &= [\widetilde{x}_1, s_3, \widetilde{s}_1]. \end{aligned}$$

Gaan na stap 5.

Stap 5:
Die huidige regterkante word gegee deur

$$\underline{b}(2) = B_2^{-1}\underline{b} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & -1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} -60 \\ 30 \\ -80 \end{bmatrix} = \begin{bmatrix} 20 \\ 50 \\ 80 \end{bmatrix}.$$

Die basisveranderlikes x_3, s_2 en $\widetilde{x}_2$ het nou die waardes 20, 50 en 80 onderskeidelik. Geen van hierdie veranderlikes kom bo hul bogrens voor nie.

Stap 6:
Al die huidige regterkante is positief, dus is 'n optimale oplossing gevind.

Die optimale oplossing word gegee deur

$$\begin{aligned} x_1 &= 80 - \widetilde{x}_1 = 80 - 0 = 80 \\ x_2 &= 80 - \widetilde{x}_2 = 80 - 80 = 0 \end{aligned}$$

$$\begin{aligned} x_3 &= 20 \\ s_1 &= 0 - \widetilde{s}_1 = 0 - 0 = 0 \\ s_2 &= 50 \\ s_3 &= 0. \end{aligned}$$

Die optimale doelfunksiewaarde is

$$\begin{aligned} \text{Doelfunksiewaarde} &= c_1x_1 + c_2x_2 + c_3x_3 \\ &= (-2)(80) + (-1)(0) + (1)(20) \\ &= -140. \end{aligned}$$

Verwysings

[1] Chartrand G & Oellermann OR, 1993, *Applied and Algorithmic Graph Theory*, McGraw Hill, Singapore.

[2] Doob M, 1996, *TEX starting from 1*, Springer-Verlag, New York.

[3] Gass SI & Harris CM, 1996, *Encyclopedia of Operations Research and Management Science*, Kluwer Academic Publishers, Boston.

[4] Hadley G, 1962, *Linear programming*, Addison-Wesley Publishing Company Inc., Reading, Massachusetts.

[5] Lamport L, 1994, *LATEX A document preparation system: User's guide and reference manual*, Addison-Wesley Longman Inc., Reading, Massachusetts.

[6] Nickolson WK, 1995, *Linear algebra with applications*, derde uitgawe, PWS Publishing Company, Boston.