Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

Het optimaliseren van een toetsenbord voor een mobiele telefoon

(Engelse titel: The optimization of a keyboard for a mobile phone)

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

JASMIJN VAN DEN BERG

Delft, Nederland
September 2016

BSc verslag TECHNISCHE WISKUNDE

"Het optimaliseren van een toetsenbord voor een mobiele telefoon"

(Engelse titel: "The optimization of a keyboard for a mobile phone")

JASMIJN VAN DEN BERG

Technische Universiteit Delft

Begeleider

Prof.dr. E. de Klerk

Overige commissieleden

Drs. E.M. van Elderen

Dr. J.L.A. Dubbeldam

September, 2016

Delft

Inhoudsopgave

Samenvatting

Omdat het veelgebruikte QWERTY-toetsenbord voor een mobiele telefoon, waarop veelal met één vinger getypt wordt, geen optimale typtijd genereert, wordt gezocht naar een nieuwe toetsenbordindeling. Om dit Toetsenbord Layout Probleem op te lossen, worden in de eerste plaats twee nieuwe toetsenroosters gecreëerd, namelijk een met hexagonale toetsen en een met rechthoekige toetsen. Met behulp van WhatsApp gesprekken en de nieuw gevormde toetsenroosters worden twee matrices gemaakt, die gebruikt worden om een kwadratisch toewijzingsprobleem (QAP) op te stellen. Vervolgens wordt met behulp van de eigenvalue bound, Gilmore-Lawler bound en reduced Gilmore-Lawler bound ondergrenzen bepaald voor dit probleem. Ook wordt de gemiddelde waarde van alle mogelijke oplossingen bepaald met de mean objective value. Hierna wordt met twee heuristieken geprobeerd voor beide toetsenrooster een optimale toetsindeling te vinden, namelijk met local search en simulated annealing. Uiteindelijk worden met simulated annealing de twee best gevonden toetsenbordindelingen gevonden. Met de hexagonale layout blijkt iets sneller te kunnen worden getypt dan met de rechthoekige layout, maar voor beide geldt dat ze maximaal 14 procent verbeterd zouden kunnen worden, kijkend naar de onderzochte ondergrenzen.

1 Het Toetsenbord Layout Probleem

In een tijd waarin alles zo efficiënt en goedkoop mogelijk moet, is het apart dat het QWERTY-toetsenbord nog steeds het meestgebruikte toetsenbord is. QWERTY is namelijk verre van optimaal op het gebied van typesnelheid, of je nu typt met '10'-vingers (in de praktijk zijn dat er meestal 9) of met 1 vinger. Maar welke layout is dan wél optimaal?

1.1 De geschiedenis van het toetsenbord

Het eerste toetsenbord, waarop door Dr. Samuel Francis patent is aangevraagd in 1857, zag eruit als een piano, met witte en zwarte toetsen. Dit toetsenbord, te zien in figuur 1, werd dan ook de Literary Piano genoemd[1].

Figuur 1: Literary Piano[1]

Figuur 2: Dvorak- en QWERTY-layout[2]

Vervolgens ontstond de typemachine, waarop de letters alfabetisch gerangschikt waren. Dit leverde echter problemen op voor de echte snelle typers. De hamertjes die de letters op het papier moesten drukken, haakten namelijk vaak in elkaar, omdat enkele veelgebruikte lettercombinaties in deze rangschikking naast elkaar lagen. Als oplossing hiervoor bedacht Christopher Sholes de QWERTY-layout (zie figuur 2). In deze layout liggen letters die veel in combinatie gebruikt worden juist ver uit elkaar [20].

Na QWERTY zijn er nog andere indelingen gekomen, waarvan Dvorak (zie figuur 2) de bekendste is. Met een Dvorak-toetsenbord kan men in het Engels sneller typen dan met QWERTY [10] en heeft men meer typecomfort. Men is echter zo gewend aan QWERTY dat dit tot op heden de meestgebruikte toetsenbordindeling is [11].

1.2 Overwegingen rondom het toetsenbord

Vroeger was de belangrijkste overweging rondom een toetsenbord dat de hamertjes van de typemachine zo min mogelijk verstrengeld zouden raken. Nu zijn er andere overwegingen die zwaar wegen in het zoeken naar een nieuwe toetsenbordlayout.

Zo moet overwogen worden of deze layout optimaal moet zijn voor '10'-vingers of voor 1 vinger. Dit wordt respectievelijk n-finger en s-finger (single-finger) genoemd. Het vinden van een optimale n-finger layout is een ontzettend ingewikkelde klus. Een optimale s-finger layout is eenvoudiger te vinden, maar ook deze vraagt veel rekentijd. Toch zijn er voor de s-finger layout heuristieken (methoden) die een relatief goede oplossing kunnen genereren; een oplossing die optimaal zou kunnen zijn. Het blijft echter lastig om aan te tonen of een gevonden oplossing de optimale is.

Waarop vooral s-finger getypt wordt, zijn de mobiele telefoon en tablet. Deze apparaten hebben te maken met beperkte ruimte voor een toetsenbord, wat resulteert in kleinere toetsen dan een computertoetsenbord. Dit heeft een voor- en een nadeel: toetsen liggen dichter bij elkaar, dus het kost minder tijd om van de ene naar de volgende toets te gaan, maar omdat de toetsen kleiner zijn, kost het 'mikken' op de juiste toets juist meer tijd. Daarnaast is er op de mobiele telefoon of tablet vaak geen ruimte voor de typische rechthoekige vorm van een toetsenbord. De ruimte die hiervoor beschikbaar is, gaat meer richting vierkant.

Wat ook een belangrijke overweging is, is de hoeveelheid karakters die meegenomen worden in de optimalisatie. Dit is nodig, zodat er een toetsenrooster bepaald kan worden, waarin alle karakters een plaats kunnen krijgen. Om een zo goed mogelijk geoptimaliseerd toetsenbord te vinden, zouden zoveel mogelijk karakters

[1] http://wonderchews.com/a-brilliant-mind/
[2] http://dvorak4tzx.lofter.com/post/1d4021c8_73b5dfe

meegenomen moeten worden in de optimalisatie, echter is het voor de rekentijd om het Toetsenbord Layout Probleem op te lossen belangrijk om een niet te groot aantal toetsen te hebben.

Een laatste en zeer belangrijke overweging, is de vraag of er wel behoefte is aan een nieuw toetsenbord. Het is immers zo dat, ondanks al bestaande betere layouts, het overgrote deel van de toetsenborden nog in QWERTY-layout is.

1.3 Dit model

De overwegingen rondom het toetsenbord meenemend, is ervoor gekozen een optimale s-finger toetsenbord layout te zoeken. Omdat de mobiele telefoon hedendaags zoveel gebruikt wordt, wordt het s-finger toetsenbord hiervoor ontworpen. Om concreet een formaat voor het toetsenbord te hebben, wordt de Nokia Lumia 520 bekeken. Deze telefoon heeft een 4 inch display, wat neerkomt op een afmeting van 5,3 × 8,7 cm [3]. Voor het gebruikersgemak wordt aangenomen dat het toetsenbord tussen een derde en de helft van het display mag innemen, dus dat is 5,3 cm breed en 2,9 tot 4,3 cm hoog.

Omdat op mobiele telefoons en tablets vooral informele teksten getypt worden, worden voor het ontwerpen van dit toetsenbord WhatsApp teksten gebruikt.

Ook is ervoor gekozen 27 karakters te optimaliseren. Het is namelijk bijna onvermijdelijk de 27 basistoetsen mee te nemen, welke de 26 letters van het alfabet bevatten en de spatie. Andere karakters, zoals de punt en komma, worden op WhatsApp beduidend minder vaak gebruikt dan in gewone teksten. Dit komt onder andere, omdat je in plaats van een punt te typen, het bericht kunt verzenden en vervolgens een nieuw bericht kunt typen. Uit onderzoek is zelfs gebleken dat het eindigen van een zin met een punt op WhatsApp vaak overkomt als minder oprecht dan als de zin zonder punt geëindigd wordt [13]. Een andere veelgebruikte toets is de shifttoets. Echter geldt ook voor de shifttoets dat deze in WhatsApp aanzienlijk minder gebruikt wordt, om tijd te besparen. Mensen begrijpen immers toch, ondanks missende hoofdletters, wat gezegd wordt. Verder is een belangrijke overweging in het optimaliseren van de toetsindeling, dat het probleem met elk extra karakter/toets aanzienlijk ingewikkelder wordt. Voor de 27 karakters die nu gekozen zijn, geldt al dat er 27! verschillende toewijzingen van karakters aan toetsen te bedenken zijn. Dit aantal wordt nog 28 keer zoveel als er slechts 1 karakter wordt toegevoegd aan de optimalisatie.

Verder is de vorm van de toetsen van belang voor het maken van een toetsenrooster. Omdat de vorm van de vingertop meer rond dan vierkant is, wordt gekozen voor een toetsenrooster met regelmatige zeshoeken. Daarnaast wordt ook een toetsenrooster met meer vierkante toetsen bekeken, omdat dat toch wat traditioneler oogt.

[3] https://www.microsoft.com/nl-nl/mobiel/support/product/lumia520/specificaties/

2 Wiskundig model: QAP

Het toewijzen van karakters aan toetsen is een kwadratisch toewijzingsprobleem (Quadratic Assignment Problem - QAP). Dit probleem is op de volgende manier weer te geven [2]:

$$z^* = \min_{\varphi \in \mathcal{S}_n} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)} + \sum_{i=1}^{n} c_{i\varphi(i)}. \tag{1}$$

Hierin zijn (veralgemeniseerd) a_{pq}, b_{pq} en c_{pq} de elementen in de p^e rij en q^e kolom van respectievelijk $(n \times n)$-matrices A, B en C. De verzameling $\mathcal{S}_n$ bevat alle mogelijke permutaties φ met $\varphi : \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$. Als dit gezien wordt als het toewijzen van n faciliteiten aan n locaties, kan de minimalisatie beter gevisualiseerd worden. Het symbool a_{ij} staat dan voor de flow van faciliteit i naar faciliteit j. Daarnaast is $b_{\varphi(i)\varphi(j)}$ de afstand van locatie $\varphi(i)$ tot locatie $\varphi(j)$. Het element $c_{i\varphi(i)}$ geeft de kosten om faciliteit i aan locatie $\varphi(i)$ toe te wijzen weer. De $a_{ij}b_{\varphi(i)\varphi(j)}$ is op te vatten als de transportkosten die gemaakt worden als faciliteit i aan locatie $\varphi(i)$ en faciliteit j aan locatie $\varphi(j)$ toegewezen worden.

Het probleem staat hier in Koopmans-Beckmann vorm en wordt QAP(A,B,C) genoemd. Het oplossen van dit probleem betekent dat de kostenfunctie z geminimaliseerd wordt, waarbij

$$z(\varphi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)} + \sum_{i=1}^{n} c_{i\varphi(i)}. \tag{2}$$

In het vervolg zal voor matrices ook een andere notatie gebruikelijk zijn, namelijk (m_{ij}) in plaats van M, voor willekeurige matrix M. Deze m_{ij} zijn de elementen van M die te vinden zijn in rij i en kolom j.

Omdat het Toetsenbord Layout Probleem een kwadratisch toewijzingsprobleem is, is het NP-moeilijk [21].

2.1 Trace-notatie

Het kwadratisch toewijzingsprobleem kan ook in trace-notatie geschreven worden. Hiervoor wordt in plaats van $b_{\varphi(i)\varphi(j)}$ te bekijken, eerst matrix B gepermuteerd. Dit gebeurt door permutatiematrix $X = (x_{ij})$ met $x_{ij} = 1$ als $j = \varphi(i)$ en anders $x_{ij} = 0$. Zo wordt $B_\varphi = XB^TX^T$ de gepermuteerde matrix B. Met deze $B_\varphi = (b_{\varphi,ij})$ zien de kosten van het QAP (zonder matrix C) er zo uit:

$$z(\varphi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi,ij}. \tag{3}$$

Nu is het zo dat deze dubbele sommatie gelijk is aan de trace van AB_φ^T. (De trace van een matrix is gelijk aan de som van de diagonale elementen.) Samen met het feit dat de tijd-afstandenmatrix B symmetrisch is en dus $B_\varphi = XB^TX^T = XBX^T$, volgt:

$$z(\varphi) = \mathrm{tr}\left(AXBX^T\right). \tag{4}$$

Het QAP kan dus eenvoudiger gesteld worden als:

$$z^* = \min_{X \in \boldsymbol{X}_n} \mathrm{tr}\left(AXBX^T\right). \tag{5}$$

Als matrix C wel meegenomen wordt, ziet de trace-notatie er overigens zo uit:

$$z^* = \min_{X \in \boldsymbol{X}_n} \mathrm{tr}\left((AXB + C)X^T\right). \tag{6}$$

2.2 Toegepast op het Toetsenbord Layout Probleem

Omdat gekozen is de optimalisatie over 27 karakters uit te voeren, geldt voor het Toetsenbord Layout Probleem dat $n = 27$ en zo ook $\varphi \in \mathcal{S}_{27}$. Matrix $A = (a_{ij})$ bevat nu de flow van karakter i naar karakter j en matrix $B = (b_{kl})$ de tijd-afstand van toets k naar toets l. Matrix $C = (c_{ik})$ heeft hier geen toegevoegde waarde, omdat het toewijzen van karakter i aan toets k voor elke i en k dezelfde kosten met zich meebrengt. Zo is $\sum_{i=1}^{27} c_{i\varphi(i)}$ een constante geworden die niet meegenomen hoeft te worden in het minimalisatieproces. Het probleem wat bekeken wordt is dus een QAP(A,B), in plaats van een QAP(A,B,C).

Het QAP bij het Toetsenbord Layout Probleem is nu op de volgende twee manieren weer te geven:

$$z^* = \min_{\varphi \in \mathcal{S}_{27}} \sum_{i=1}^{27} \sum_{j=1}^{27} a_{ij} b_{\varphi(i)\varphi(j)} \tag{7}$$

of

$$z^* = \min_{X \in \boldsymbol{X}_{27}} \operatorname{tr}(AXBX^T). \tag{8}$$

3 Technieken voor QAP

Er zijn verschillende technieken ontwikkeld die gebruikt kunnen worden om een QAP op te lossen of de optimale oplossing te benaderen. In dit hoofdstuk worden daarvan enkele behandeld.

3.1 Eigenvalue bound

De eigenvalue bound geeft een ondergrens voor de optimale oplossing. Hiervoor wordt aangenomen dat $A = A^T$ en $B = B^T$. Omdat A en B symmetrisch zijn, zijn deze twee matrices ook diagonaliseerbaar en kunnen ze dus geschreven worden in spectraalontbinding als

$$A = \sum_{i=1}^{n} \lambda_i p_i p_i^T, \quad B = \sum_{j=1}^{n} \mu_j q_j q_j^T, \tag{9}$$

waarin p_i en q_j orthonormale eigenvectoren en λ_i en μ_j eigenwaarden zijn van respectievelijk matrix A en B.

Voor $\mathrm{tr}(AB)$ geldt met (9) het volgende:

$$\begin{aligned} \mathrm{tr}(AB) &= \mathrm{tr}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \mu_j p_i p_i^T q_j q_j^T\right) && (10)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \mu_j \langle p_i, q_j\rangle \mathrm{tr}(p_i q_j^T) && (11)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \mu_j \langle p_i, q_j\rangle^2 && (12) \end{aligned}$$

Van (10) naar (11) worden alle scalairen buiten de trace gehaald, waaronder $p_i^T q_j$, wat gelijk is aan het inproduct $\langle p_i, q_j\rangle$ en zodanig een scalair is. Van (11) naar (12) wordt gebruik gemaakt van het feit dat ook $\mathrm{tr}(p_i q_j)$ gelijk is aan $\langle p_i, q_j\rangle$.

Omdat $p_1 \ldots p_n$ en $q_1 \ldots q_n$ orthogonale bases vormen voor $\mathbb{R}^n$. geldt dat

$$\sum_{i=1}^{n} \langle p_i, q_j\rangle^2 = \langle q_j, q_j\rangle = 1, \quad \sum_{j=1}^{n} \langle p_i, q_j\rangle^2 = \langle p_i, p_i\rangle = 1. \tag{13}$$

Hieruit volgt dat matrix $S = (\langle p_i, q_j\rangle^2)$ dubbel stochastisch is.

Stelling 3.1 (Birkhoff [1], 1946.)**.** *The knopen van de toewijzingspolytoop komen uniek overeen met permutatie matrices.*

Volgens deze stelling van Birkhoff kan S dus geschreven worden als convexe combinatie van de permutatiematrices X_r:

$$S = \sum_r \alpha_r X_r, \quad \forall r : X_r \in \boldsymbol{X}_n, \quad \alpha_r \geq 0; \quad \sum_r \alpha_r = 1. \tag{14}$$

Hiermee volgt dat

$$\begin{aligned} \mathrm{tr}(AB) &= \sum_r \alpha_r \langle \lambda, X_r \mu\rangle && (15)\\ &\geq \min_{\varphi\in\mathcal{S}_n} \sum_{i=1}^{n} \lambda_i \mu_{\varphi(i)} && (16)\\ &=: \langle \lambda, \mu\rangle^-. && (17) \end{aligned}$$

De definitie van $\langle \lambda, \mu\rangle^-$ volgt uit de volgende propositie:

Propositie 3.1 (Hardy, Littlewood & Pólya [15], 1952)**.** *Zij $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ en $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$. Dan geldt voor elke permutatie φ*

$$\sum_{i=1}^{n} \lambda_i \mu_{n+1-i} \leq \sum_{i=1}^{n} \lambda_i \mu_{\varphi(i)} \leq \sum_{i=1}^{n} \lambda_i \mu_i$$

Gevolg 3.1.1. *Als* $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ *en* $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_n$, *dan* $\forall \varphi$

$$\langle \lambda, \mu \rangle^- = \sum_{i=1}^{n} \lambda_i \mu_i \leq \sum_{i=1}^{n} \lambda_i \mu_{\varphi(i)}$$

Er is nu aangetoond dat $tr(AB) \geq \langle \lambda, \mu \rangle^-$ en omdat voor elke $X \in \boldsymbol{X}_n$ geldt dat $B \sim XBX^T$, omdat beide matrices eigenwaarde μ hebben, kan B vervangen worden door XBX^T. Zo volgt dus

Propositie 3.2.

$$z^* = \min_{X \in \boldsymbol{X}_n} tr(AXBX^T) \geq \langle \lambda, \mu \rangle^-$$

.

3.2 Gilmore-Lawler bound

Ook de Gilmore-Lawler bound (zie bijvoorbeeld [4]) geeft een ondergrens voor de optimale oplossing. Voor deze bound moeten matrix A en B niet-negatief zijn en zijn voor beide matrices alle elementen op de diagonaal 0. Zonder verlies van algemeenheid kan dit voor alle matrices aangenomen worden. Bij matrices met negatieve elementen kan immers αJ opgeteld worden, met α de positieve waarde van het kleinste element en J de matrix waarvan elk element 1 is. Dit voegt na uitwerken van haakjes slechts een constante toe, die geen invloed heeft op de minimalisatie. Daarnaast kunnen de diagonale elementen naar matrix C verplaatst worden.

Het idee van de Gilmore-Lawler bound is nu om het kwadratische deel te lineariseren, zodat er slechts een perfecte bipartiete matching van minimaal gewicht gevonden hoeft te worden. Hierbij wordt dus gekeken naar

$$z(\varphi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)}. \tag{18}$$

Voor elke vaste i kan gesteld worden dat

$$\sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)} \geq \langle \hat{a}_i, \hat{b}_{\varphi(i)} \rangle^-, \tag{19}$$

waarbij $\hat{a}_i$ en $\hat{b}_{\varphi(i)}$ vectoren zijn die alle elementen van rij a_i respectievelijk $b_{\varphi(i)}$ bevatten, behalve de diagonale elementen a_{ii} en $b_{\varphi(i)\varphi(i)}$. zodat volgt dat voor alle φ

$$z(\varphi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\varphi(i)\varphi(j)} \geq \sum_{i=1}^{n} \langle \hat{a}_i, \hat{b}_{\varphi(i)} \rangle^- \tag{20}$$

Praktisch wordt dit lineariseren en het vervolgens matchen op de volgende manier uitgevoerd:

- Zij $\hat{a}_i \in \mathbb{R}^{n-1}$ rijvectoren die de elementen van rij a_i bevat, behalve het element a_{ii}. Zij ook zo de vectoren $\hat{b}_i$ vanuit matrix B.
- Zij $L = \left(\langle \hat{a}_i, \hat{b}_j \rangle^-\right) + C + (a_{ii} b_{jj})$.
- Pas het Hongaarse algoritme [18] toe op matrix $L = (\ell_{ij})$ om $\ell^* = \min_{\varphi \in \mathcal{S}_n} \sum_{i=1}^{n} \ell_{i\varphi(i)}$ te bepalen.

Deze ℓ^* is een ondergrens voor het QAP [22], dus

$$z^* \geq \ell^*. \tag{21}$$

3.3 Reduced Gilmore-Lawlerbound

De eerste die een verbeterde Gilmore-Lawlerbound introduceerde was Conrad [9] in 1971. Sindsdien hebben verschillende wiskundigen toevoegingen gedaan. Het algemene idee in al deze verbeterde Gilmore-Lawlerbounds is om de kwadratische termen te lineariseren door ze te splitsen in lineaire termen. Voor het

QAP (waarvan beide matrices A en B alle diagonale elementen nul hebben) worden de kwadratische termen op de volgende manier ontbonden:

$$a_{ik} = \overline{a}_{ik} + r_k + u_i \quad (i,k = 1,2,\ldots,n; i \neq k) \tag{22}$$
$$b_{jl} = \overline{b}_{jl} + s_l + v_j \quad (j,l = 1,2,\ldots,n; j \neq l) \tag{23}$$

met reële vectoren $r,\ s,\ u,\ v \in \mathbb{R}^n$.

Frieze en Yadegar [12] hebben laten zien dat vector u en v geen effect hebben op de waarde van de ondergrens, dus weggelaten kunnen worden. Nu kan dus vanuit (22) en (23) gesteld worden dat

$$\overline{a}_{ik}\overline{b}_{jl} = (a_{ik} - r_k)(b_{jl} - s_l) = a_{ik}b_{jl} - r_k b_{jl} - s_l a_{ik} + r_k s_l. \tag{24}$$

Nu kan dus gesteld worden dat

$$\sum_{i=1}^{n}\sum_{k=1}^{n} a_{ik}b_{\varphi(i)\varphi(k)} + \sum_{k=1}^{n} c_{k\varphi(k)} = \sum_{i=1}^{n}\sum_{k=1}^{n} \overline{a}_{ik}\overline{b}_{\varphi(i)\varphi(k)} + \sum_{k=1}^{n} \overline{c}_{k\varphi(k)} \tag{25}$$

met

$$\overline{c}_{kl} = c_{kl} + r_k \sum_{j=1}^{n} b_{jl} + s_l \sum_{i=1}^{n} a_{ik} - (n-1) r_k s_l \tag{26}$$

Hierbij worden r_k en s_l op de volgende manier gekozen, zodat $\overline{a}_{ik}$ en $\overline{b}_{jl}$ niet-negatief zijn:

$$r_k = \min\{a_{ik} : 1 \leq i \leq n; i \neq k\} \quad (k = 1,2,\ldots,n), \tag{27}$$
$$s_l = \min\{b_{jl} : 1 \leq j \leq n; j \neq l\} \quad (l = 1,2,\ldots,n). \tag{28}$$

Aangezien $\overline{a}_{ik} = a_{ik} - r_k$ en $\overline{b}_{jl} = b_{jl} - s_l$ geldt, kunnen matrices $\overline{A}$ en $\overline{B}$ bepaald worden. Ook kan met (26)-(28) matrix $\overline{C} = (\overline{c}_{kl})$ gevonden worden. Met deze drie matrices kan nu de Gilmore-Lawlerbound berekend worden, wat opnieuw een ondergrens oplevert voor het kwadratisch toewijzingsprobleem [12].

3.4 Mean objective value

Het kan interessant zijn om de gemiddelde kosten over alle mogelijke permutaties te bepalen, om op deze manier te bepalen hoeveel beter een bepaalde oplossing is dan de gemiddelde waarde. Hiervoor kan de mean objective value bepaald worden met behulp van kansrekening.

Voor de bepaling van de mean objective value, wordt eerst matrix C geherdefinieerd als $C = (c_{ij} + a_{ii}b_{jj})$, zodat matrix A en B alle diagonale elementen nul hebben.

Voor het toewijzen van een karakter i aan toets j is het eenvoudig te zien dat

$$P(i \to j) = \frac{1}{n} \quad \forall i,j \tag{29}$$

en de kans dat ook karakter k aan toets l toegewezen wordt, is nu

$$P((i \to j) \wedge (k \to l)) = \frac{1}{n(n-1)} \quad \text{voor } i \neq k,\ j \neq l. \tag{30}$$

Zo wordt de mean objective value gegeven door

$$\mu(A,B,C) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{ik}b_{jl}P((i \to j) \wedge (k \to l)) + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}P(i \to j), \tag{31}$$

oftewel [3]

$$\mu(A,B,C) = \frac{1}{n(n-1)} \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{ij}b_{kl} + \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}. \tag{32}$$

3.5 Local search - k-exchange neighborhood

Een heuristiek om een goede oplossing voor het QAP te vinden, is de local search. Bij local search wordt vanuit een willekeurige oplossing φ de neighborhood verkend. Deze k-exchange neighborhood bevat alle oplossing φ' die maximaal k verwijderd liggen van de willekeurig gekozen oplossing φ. Deze afstand ziet er wiskundig zo uit [6]:

$$d(\varphi, \varphi') = \left|\{i : \varphi(i) \neq \varphi'(i)\}\right| \tag{33}$$

Hiermee wordt de k-exchange neighborhood

$$\mathcal{N}_k(\varphi) = \{\varphi' \in \mathcal{S} : d(\varphi, \varphi') \leq k\} \quad (\text{met } 2 \leq k \leq n). \tag{34}$$

De heuristiek is nu eenvoudigweg:

1. Zij $\varphi \in \mathcal{S}_n$ de huidige oplossing.
2. Zoek een $\varphi' \in \mathcal{N}_k(\varphi)$ zó dat $z(\varphi') < z(\varphi)$ en vervang φ door φ'.
3. Herhaal dit.

Op het moment dat er geen betere oplossing in de k-exchange neighborhood te vinden is, is er een lokaal minimum bereikt. Dit hoeft dus niet de optimale oplossing te zijn.

3.6 Simulated annealing

De heuristiek simulated annealing maakt ook gebruik van neighborhoods, maar accepteert niet alleen betere oplossingen. Met een bepaalde kans neemt deze heuristiek ook slechtere oplossingen aan, zodat er niet in lokale minima blijven hangen wordt. Deze procedure komt voort uit de thermodynamica en de metallurgie en maakt gebruik van het gegeven dat bij langzaam afkoelen van gesmolten metalen, de metalen mooi kristalliseren [17]. Bij simulated annealing wordt op dezelfde manier gebruik gemaakt van een dalende temperatuur. Deze temperatuur wordt gekoppeld aan de kans dat een slechtere oplossing toch aangenomen wordt: bij een lagere temperatuur, wordt minder snel slechtere oplossing geaccepteerd [8]. Hierdoor ziet het algoritme voor simulated annealing er op de volgende manier uit, met beginwaarden $\varphi = (1, \ldots, n)$, $t = 1$ en T vast gekozen, 'groot genoeg' [16]:

1. Zij de huidige oplossing $\varphi \in \mathcal{S}_n$, de huidige iteratie t en de huidige temperatuur T.
2. Als $2n$ een deler is van t, zet $T \to T/2$ [7].
3. Kies $\varphi' \in \mathcal{N}_k(\varphi)$ willekeurig.
4. Vervang φ door φ' met kans

$$\mathbb{P}[\varphi' \text{ wordt geaccepteerd}] = \begin{cases} 1 & \text{als } z(\varphi') \leq z(\varphi) \\ \exp\left(\dfrac{z(\varphi) - z(\varphi')}{T}\right) & \text{als } z(\varphi') > z(\varphi) \end{cases} \tag{35}$$

5. Zet $t \to t + 1$ en herhaal.

Simulated annealing is een sterkere heuristiek dan local search. Dat volgt uit de volgende stelling:

Stelling 3.2 (Hajek [14], 1988)**.** *Voor de iteraties t van simulated annealing met de k-exchange neighborhood geldt*

$$\lim_{t \to \infty} \mathbb{P}[\varphi_t \textit{ is optimaal }] = 1$$

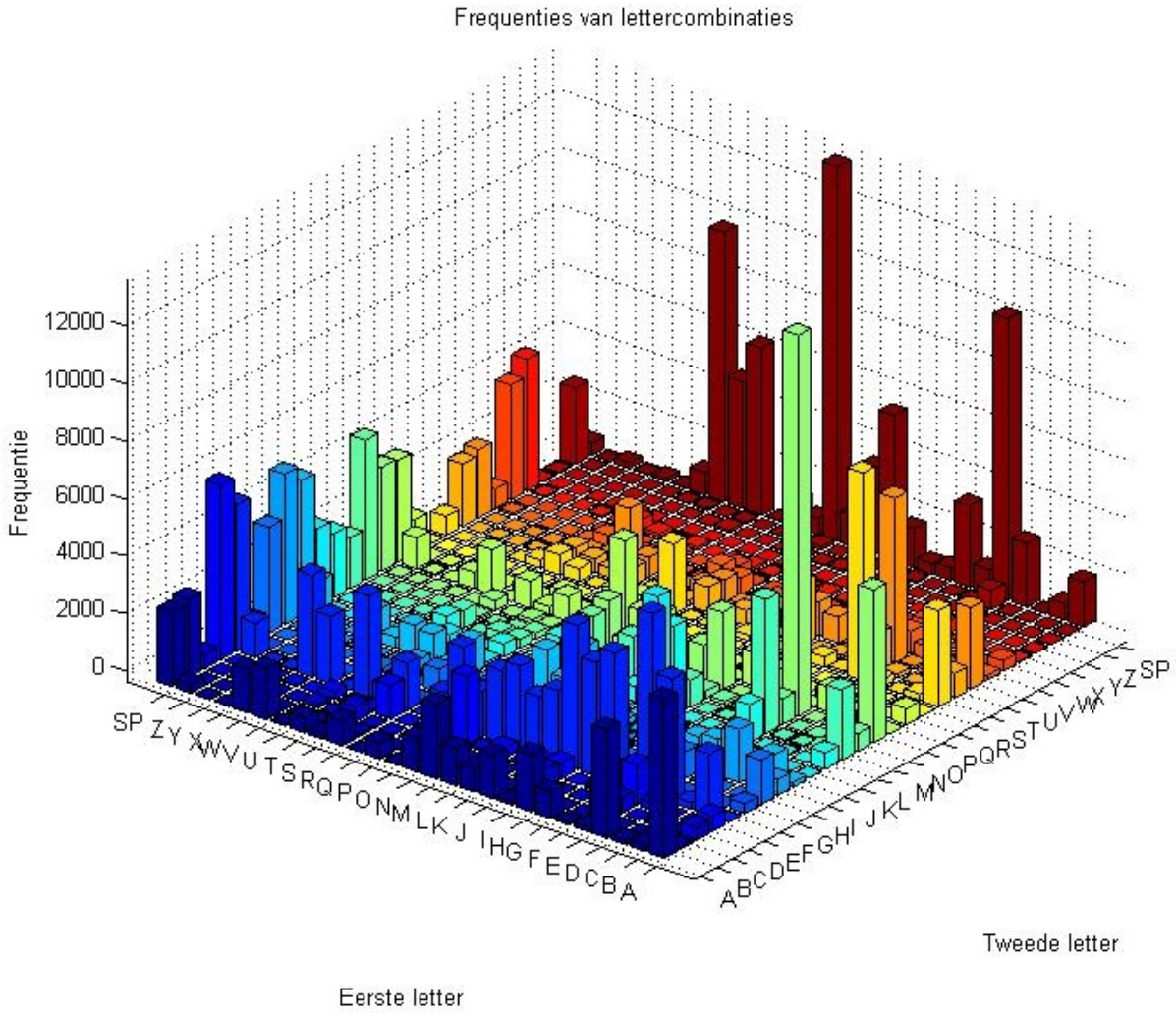
Dit betekent niet dat het vinden van een optimale oplossing met deze methode gegarandeerd is, omdat de limiet buiten de kans staat. Het staat echter voor een convergentie in waarschijnlijkheid dat een gevonden φ de optimale oplossing is.

4 Genereren van QAP data

Om de optimalisatie uit te voeren, is data nodig in de vorm van matrices A en B. In het komende hoofdstuk zal beschreven worden hoe deze data is verkregen en verwerkt tot matrices.

4.1 Frequenties van lettercombinaties - Matrix $\mathbf{A}_{sym}$

Om te weten welke karakters dicht bij elkaar moeten komen op het toetsenbord, is het van belang te weten hoe vaak elke lettercombinatie voorkomt. Hiervoor wordt een (27×27)-matrix A in het leven geroepen, die op plek (1,2) de frequentie van de combinatie 'ab' heeft staan (a is immers de 1e letter en b de 2e), op dezelfde manier op plek (11,5) de frequentie van de combinatie 'ke' en op plek (14,27) de frequentie van de combinatie 'n∅' (het symbool ∅ wordt in het vervolg gebruikt om een spatie aan te geven).
Om een beeld te krijgen van de inhoud van deze matrix, die u ook in bijlage 7.1.1 kunt vinden, is hier een 3D-staafdiagram van gemaakt die u kunt vinden in figuur 3. Hoe deze data verzameld is, leest u in paragraaf 4.1.2.



Figuur 3: De frequenties van lettercombinaties

4.1.1 Symmetrisch maken

Voor veel technieken om de oplossing van het QAP te vinden is het nodig dat matrix A symmetrisch is. Hiertoe wordt matrix A_{sym} gedefinieerd:

$$A_{sym} := \frac{1}{2}\left(A + A^T\right). \tag{36}$$

Het gebruik van deze matrix A_{sym} levert dezelfde oplossing. Dit is eenvoudig met de axioma's van de trace af te leiden:

$$\begin{aligned} \mathrm{tr}(\frac{1}{2}(A + A^T)XBX^T) &= \frac{1}{2}\mathrm{tr}(AXBX^T) + \frac{1}{2}\mathrm{tr}(A^TXBX^T) && (37)\\ &= \frac{1}{2}\mathrm{tr}(AXBX^T) + \frac{1}{2}\mathrm{tr}(XB^TX^TA) && (38)\\ &= \frac{1}{2}\mathrm{tr}(AXBX^T) + \frac{1}{2}\mathrm{tr}(AXBX^T) && (39)\\ &= \mathrm{tr}(AXBX^T). && (40) \end{aligned}$$

Van (37) naar (38) is hierbij gebruik gemaakt van de sokken-schoenenregel en het feit dat voor de trace geldt dat $\mathrm{tr}(M) = \mathrm{tr}(M^T)\ \forall M \in \mathbb{R}^{n\times n}$. Uit (38) volgt (39), omdat $B = B^T$ gegeven is en omdat geldt dat $\mathrm{tr}(MNOP) = \mathrm{tr}(PMNO)\ \forall M, N, O, P \in \mathbb{R}^{n\times n}$.

4.1.2 Data

Op de mobiele telefoon wordt vooral getypt in WhatsApp. In deze app is het taalgebruik vaak anders dan de taal die gebruikt wordt bij het typen van verslagen of artikelen, wat vaak op de computer gebruikt. Om deze reden bestaat alle data die gebruikt wordt voor de optimalisatie, uit WhatsApp gesprekken. Er is geprobeerd om een breed scala aan gesprekken te bemachtigen, met zowel mannen als vrouwen uit diverse delen van het land en van diverse leeftijden.

4.2 Toetsenrooster - Matrix B

Er moet nog bepaald worden welke vorm en opstelling de toetsen zullen hebben. De verschillende mogelijkheden, waarbij een gevuld raster gemaakt kan worden, zijn regelmatige driehoekige, vierkante of zeshoekige toetsen. Onmiddellijk kunnen regelmatige driehoeken als toetsen uitgesloten worden, omdat dit totaal geen gebruiksgemak geeft. De gemiddelde vinger is immers niet driehoekig van vorm. Omdat zeshoekige toetsen het meest in de buurt komen van de vorm van de vinger, is een toetsenrooster opgebouwd uit regelmatige zeshoeken het uitgangspunt. Daarnaast wordt een toetsenrooster met rechthoekige toetsen bekeken, omdat dat toch de meer traditionele vorm is.

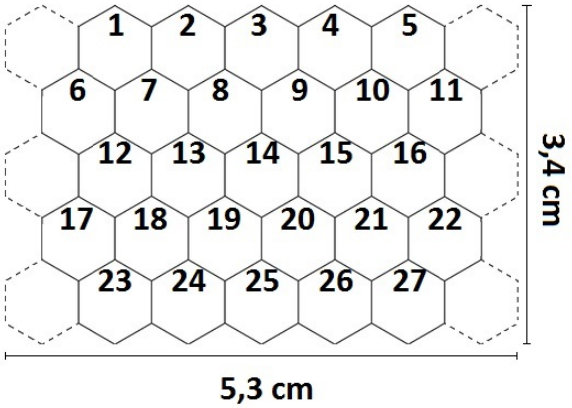
Zo komt de matrix B in twee versies voor, namelijk matrix $B6$ voor het hexagonale toetsenrooster en matrix $B4$ voor het rechthoekige toetsenrooster.

4.2.1 Toetsenrooster - Matrix B6

In figuur 4 ziet u het hexagonale toetsenrooster. Dit rooster is volledig opgebouwd uit regelmatige zeshoeken. Merk op dat dit rooster 27 toetsen bevat, aangezien er ook 27 karakters bekeken worden, maar daarnaast ook zes additionele toetsen, die bijvoorbeeld voor de shift, backspace, komma of punt gebruikt kunnen worden. Omdat de breedte van een Nokia Lumia 520 vastligt op 5,3 cm, kan hiermee de hoogte van dit toetsenrooster bepaald worden op 3,4 cm. De breedte van een toets is dus 0,76 cm.

Vervolgens wordt er een matrix $B6$ (zie bijlage 7.1.3) opgesteld, waarin de tijd-afstanden tussen toetsen opgeslagen wordt. Deze tijd-afstanden geven de tijd weer die het (gemiddeld) kost om van de ene toets naar de andere te bewegen. Deze tijd is niet alleen afhankelijk van de afstand A tussen de middelpunten van deze twee toetsen, maar ook van de diameter D van de toets in de bewegingsrichting [19]. Om deze tijd-afstand te berekenen wordt Fitts' law gebruikt. Deze wet zegt dat voor de bewegingstijd MT (movement time) geldt:

$$MT = \alpha + \beta \log_2\left(\frac{A}{D} + 1\right) \tag{41}$$



Figuur 4: De layout van een toetsenbord met zeshoekige toetsen

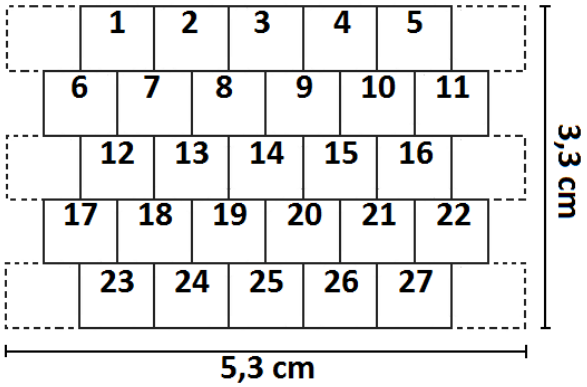
Hierin zijn α en β constanten, A is de afstand tussen de middelpunten van twee toetsen en D is de diameter die, zoals zojuist beschreven, gemeten wordt in de bewegingsrichting.

Merk hierbij op dat het achtereenvolgens meerdere keren voorkomen van hetzelfde karakter geen invloed heeft op de toetsenbordindeling. Er wordt immers geen afstand afgelegd, dus geldt $MT = \alpha$. Hieruit volgt dat de locatie van het dubbelgebruikte karakter niet uit maakt. Om deze reden kan gesteld worden dat in matrix A $a_{ii} = 0 \; \forall i$.

Voor α en β wordt gekozen dat $\alpha = 0$ en $\beta = \frac{10}{49}$ [11]. De functie die de movement time berekent, kunt u vinden in bijlage 7.2.1. Met deze MT kan nu voor elk element $b_{\varphi(i)\varphi(j)}$ de waarde bepaald worden, door de afstand tussen de middelpunten van toets $\varphi(i)$ en toets $\varphi(j)$ en de diameter van toets $\varphi(j)$ in de bewegingsrichting te bepalen. In bijlage 7.1.3 vindt u deze matrix $B6$.

4.2.2 Toetsenrooster - Matrix B4

De zeshoeken van bovenstaand toetsenrooster kunnen eenvoudig veranderd worden in meer traditionele vierkante of rechthoekige toetsen. Om zo dicht mogelijk bij het ontwerp met de zeshoekige toetsen te blijven, liggen de middelpunten van de rechthoekige toetsen op dezelfde afstand van elkaar als de middelpunten van de zeshoekige toetsen. In figuur 5 ziet u hoe dit toetsenrooster er uit ziet. Ook deze layout bevat 27 toetsen, met zes extra toetsen die niet meegenomen worden in de optimalisatie. De toetsen hebben een afmeting van $0,76 \times 0,66$ cm. Zoals voor matrix $B6$, wordt ook voor matrix $B4$ Fitts' law toegepast. Zie bijlage



Figuur 5: De layout van een toetsenbord met rechthoekige toetsen

7.1.4 voor de matrix $B4$ die hieruit volgt.

5 Resultaten

5.1 Ondergrenzen en mean objective value

De resultaten van de eigenvalue bound, Gilmore-Lawler bound, reduced Gilmore-Lawler bound en mean objective value (zie paragraaf 3.1-3.4) voor beide toetsenroosters vindt u in figuur 6. Hierbij is gebruik gemaakt van matrices A_{sym} en $B6$ (voor het hexagonale toetsenrooster) of $B4$ (voor het rechthoekige toetsenrooster). De Matlabcodes kunt u vinden in bijlagen 7.3.1-7.3.4. Merk op dat de drie gebruikte matrices slechts nullen op de diagonaal hebben, waardoor voldoet wordt aan de voorwaarden voor de vier bovengenoemde methoden.

	Hexagonaal	Rechthoekig
Eigenvalue bound	−196058	−192440
Gilmore-Lawler bound	88006	88523
Reduced Gilmore-Lawler bound	88045	88562
Mean objective value	130350	130752

Figuur 6: Resultaten ondergrenzen

Zoals u kunt zien zijn de eigenvalue bounds voor beide toetsenroosters negatief en dus niet relevant, aangezien de kosten al nooit kleiner dan 0 konden worden.

De Gilmore-Lawler bound en de reduced Gilmore-Lawler bound liggen qua waarden erg dicht bij elkaar, waarbij de reduced slechts verwaarloosbaar beter is.

5.2 Local search - k-exchange neighborhood

In paragraaf 3.5 heeft u al kunnen lezen hoe de heuristiek local search een lokaal minimum kan vinden voor het Toetsenbord Layout Probleem. Nu zijn er echter nog verschillende manieren om een k-exchange neighborhood te bepalen en te beslissen of je een betere oplossing aanneemt of niet.

De eerste mogelijkheid is om Matlab random k unieke getallen uit $\{1, \ldots, 27\}$ te laten genereren en als nieuwe permutatie φ de permutatie te nemen waarbij de verwisseling van de k karakters de meest gunstige kosten oplevert. De Matlabcode die gebruikt is voor deze heuristiek staat in bijlage 7.3.5, waarbij de kostenfunctie wordt gebruikt die u kunt vinden in bijlage 7.2.2. In de figuren 7 en 8 vindt u de laagst gevonden kosten met bijbehorende permutatie φ bij gebruik van de random local search heuristiek met zowel de 2-exchange neighborhood als de 3-exchange neighborhood. Ook zijn er verschillende aantallen iteraties t gebruikt. Hierbij is in alle situaties als beginsituatie $\varphi(i) = i,\ \forall i$ genomen.

k	t	$z(\varphi)$	φ
2	1000	102467	(16, 18, 5, 21, 14, 12, 10, 11, 19, 24, 25, 13, 4, 20, 3, 7, 17, 8, 26, 9, 27, 2, 22, 6, 1, 23, 15)
2	10000	101971	(7, 17, 22, 3, 13, 1, 12, 21, 9, 4, 10, 2, 25, 8, 19, 23, 11, 18, 15, 20, 16, 24, 6, 5, 27, 26, 14)
2	100000	101701	(8, 27, 4, 13, 20, 17, 25, 3, 21, 22, 16, 26, 7, 15, 18, 2, 5, 19, 10, 9, 11, 12, 24, 1, 6, 23, 14)
3	1000	101656	(13, 11, 2, 8, 15, 23, 26, 7, 16, 22, 21, 10, 18, 20, 25, 12, 6, 19, 3, 9, 5, 24, 4, 1, 17, 27, 14)
3	10000	101601	(13, 11, 23, 8, 15, 5, 4, 18, 21, 26, 10, 16, 7, 20, 3, 12, 6, 9, 25, 19, 24, 2, 22, 17, 1, 27, 14)

Figuur 7: Resultaten random local search voor hexagonale toetsenrooster

k	t	$z(\varphi)$	φ
2	1000	102683	(19, 4, 26, 18, 8, 23, 2, 25, 9, 10, 15, 3, 24, 13, 12, 6, 11, 7, 21, 20, 16, 17, 1, 22, 27, 5, 14)
2	10000	102658	(13, 22, 2, 8, 15, 27, 18, 7, 10, 4, 16, 21, 12, 9, 25, 23, 6, 19, 26, 20, 3, 24, 5, 1, 17, 11, 14)
2	100000	102046	(13, 22, 12, 19, 15, 23, 21, 7, 10, 4, 5, 16, 18, 9, 25, 26, 17, 20, 3, 8, 11, 24, 2, 6, 1, 27, 14)
3	1000	102045	(15, 17, 16, 20, 13, 27, 18, 10, 7, 2, 1, 12, 21, 8, 25, 24, 11, 19, 3, 9, 6, 26, 4, 5, 22, 23, 14)
3	10000	102225	(12, 21, 1, 18, 14, 16, 24, 6, 9, 3, 4, 15, 17, 8, 20, 26, 22, 19, 2, 7, 10, 25, 23, 11, 27, 5, 13)

Figuur 8: Resultaten random local search voor rechthoekige toetsenrooster

Een andere manier om local search uit te voeren, is om in elke iteratie op gestructureerde wijze alle 27^k mogelijke k-exchange neighborhoods te bekijken en vervolgens de permutatie aan te nemen die de meest

gunstige kostenbesparing oplevert. Zogauw er in een iteratie geen verbetering meer optreedt, stopt het algoritme. Het enige wat bij dit algoritme invloed heeft op de gevonden eindwaarde, is de beginpermutatie φ_b. Om deze reden worden enkele verschillende φ_b's bekeken. De matlabcode voor de gestructureerde local search kunt u vinden in bijlage 7.3.6 en de resultaten in figuur 9 en 10.

k	φ_b	$z(\varphi)$	φ
2	$(1:27)$	103123	$(3,24,6,7,13,17,2,12,15,16,21,18,4,8,10,11,27,9,20,19,26,5,25,1,23,22,14)$
2	$(14:27,1:13)$	101897	$(18,6,4,25,13,27,12,3,20,21,15,24,2,19,8,1,11,7,10,9,16,17,23,22,5,26,14)$
3	$(1:27)$	103242	$(19,1,27,18,8,11,2,26,9,4,10,3,21,13,15,24,17,7,25,20,6,16,12,23,22,5,14)$
3	$(14:27,1:13)$	102364	$(7,26,4,12,19,27,17,3,24,23,25,20,2,18,9,10,22,8,15,14,21,1,6,11,5,16,13)$

Figuur 9: Resultaten gestructureerde local search voor hexagonale toetsenrooster

k	φ_b	$z(\varphi)$	φ
2	$(1:27)$	102874	$(8,23,4,9,19,27,12,3,15,16,10,25,2,13,7,17,22,18,21,20,26,6,24,11,5,1,14)$
2	$(14:27,1:13)$	103756	$(12,1,16,7,14,5,20,21,18,23,24,17,6,19,15,4,22,9,3,8,10,2,25,27,11,26,13)$
3	$(1:27)$	102928	$(7,17,26,3,13,23,18,25,9,4,10,12,2,8,15,24,11,19,21,20,16,6,1,22,27,5,14)$
3	$(14:27,1:13)$	103372	$(7,10,16,12,19,27,6,15,24,23,17,25,2,18,9,1,11,8,20,14,21,3,26,22,5,4,13)$

Figuur 10: Resultaten gestructureerde local search voor rechthoekige toetsenrooster

5.3 Simulated annealing

Ook de simulated annealing, waarover te lezen is in paragraaf 3.6, is op twee manieren uitgevoerd, namelijk random en gestructureerd.

Hierbij is voor de hexagonale layout het beste resultaat gevonden bij zowel random simulated annealing met $k = 2$ en $T_{begin} = 10000$, als met de gestructureerde versie. De kosten die in beide gevallen gevonden zijn, zijn $z(\varphi) = 101499$. De Matlabcodes die hiervoor gebruikt zijn, vind u in bijlage 7.3.7 en 7.3.8 voor random simulated annealing respectievelijk gestructureerde simulated annealing.

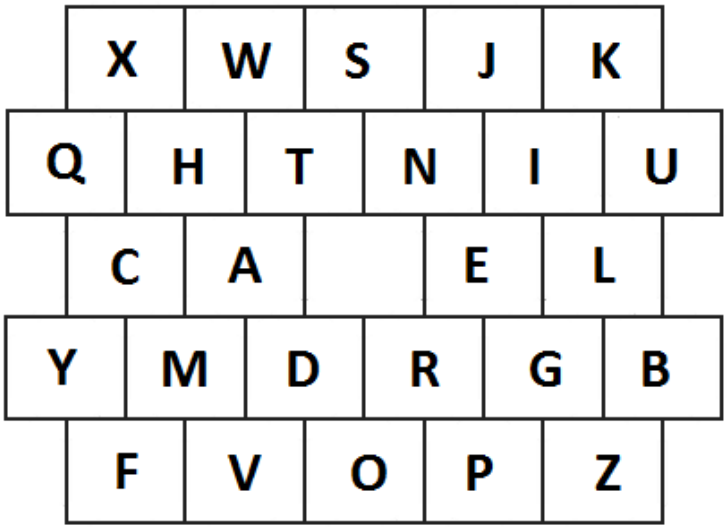
Voor de rechthoekige layout is het beste resultaat gevonden met random simulated annealing met $k = 3$ en $T_{begin} = 5000$. De gevonden kosten zijn $z(\varphi) = 102192$.

Daarnaast is er een Matlabcode geschreven, die u kunt vinden in bijlage 7.3.9, waarin de gestructureerde simulated annealing herhaaldelijk wordt uitgevoerd, tot er een oplossing gevonden wordt die beter is dan een zelf gekozen waarde. Dit is een behoorlijk doelgerichte code, die dus alleen een oplossing geeft als er een betere oplossing te vinden is dan de tot dan toe bekende. De oplossingen die uiteindelijk, na lange tijd runnen van deze code, de beste bleken te zijn, kunt u vinden in figuur 11. In 12 voor de hexagonale layout en 13 voor de rechthoekige layout vindt u de uiteindelijke layouts die bij de best gevonden oplossingen horen.

	$z(\varphi)$	φ
hexagonaal	101499	$(13,22,1,19,15,27,21,7,10,5,4,16,18,9,25,12,17,20,3,8,2,24,26,6,23,11,14)$
rechthoekig	102045	$(13,22,12,19,15,23,21,7,10,4,5,16,18,9,25,26,6,20,3,8,11,24,2,1,17,27,14)$

Figuur 11: Beste resultaat Toetsenbord Layout Probleem voor hexagonale en rechthoekige layout

Figuur 12: Best gevonden layout voor hexagonale toetsen, $z = 101499$.

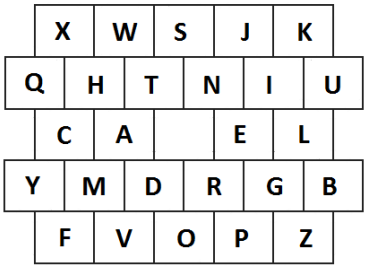


Figuur 13: Best gevonden layout voor rechthoekige toetsen, $z = 102045$.

6 Conclusie en aanbevelingen

De best gevonden toetsenborden voor de mobiele telefoon in de Nederlandse taal vindt u in figuur 14 en 15. Deze oplossingen zijn beide gevonden met behulp van simulated annealing. De toetsenbordindeling van de rechthoekige layout was ook al (in spiegelbeeld) gevonden met behulp van local search. Om het verschil tussen de toetsenbordindelingen van de twee verschillende layouts duidelijk te kunnen zien, is ervoor gekozen beide indelingen zo te spiegelen, dat linksboven de spatie de letter 'T' te vinden is en rechtboven de spatie de letter 'N'.

Figuur 14: Best gevonden layout voor hexagonale toetsen, $z = 101499$.



Figuur 15: Best gevonden layout voor rechthoekige toetsen, $z = 102045$.

Voor de hexagonale layout met kosten $z = 101499$ geldt dat deze toetsenindeling niet meer dan 14 procent beter kan dan deze z, vanwege de ondergrens $\underline{z} = 88045$ die volgt uit de reduced Gilmore-Lawler bound (zie figuur 6).

Ook de rechthoekige layout, met best gevonden kosten $z = 102045$, kan niet meer dan 14 procent verbeterd worden, vanwege de Gilmore-Lawler bound die de ondergrens $\underline{z} = 88562$ geeft.

Met deze informatie ligt de voorkeur voor een layout bij de hexagonale versie. Deze versie heeft namelijk iets lagere kosten, ook al is dat verschil te verwaarlozen.

Het zou zeker aan te raden zijn om te onderzoeken, bijvoorbeeld met behulp van branch-and-bound [5], of er betere ondergrenzen te vinden zijn, met als doel de bovengenoemde percentages te verkleinen en het op die manier aannemelijker te maken dat de hierboven gevonden toetsenborden optimale toetsenborden zijn. Zo gauw er een ondergrens gevonden wordt, waardoor de best gevonden kosten nog slechts 5 procent verbeterd zouden kunnen worden, is deze aannemelijkheid een stuk verhoogd en is het bijna de moeite niet waard om te onderzoeken of de gevonden oplossing de beste is of niet.

Mocht er bewezen worden dat deze toetsenbordindelingen binnen 5 procent van een betere ondergrens dan tot nu toe gevonden liggen, kan overwogen worden de toetsenbordindelingen op de markt te brengen. Hierbij spelen echter enkele praktische overwegingen mee, die in de volgende alinea's doorlopen worden.

De eerste overweging is de reactie van de Nederlander op een nieuw toetsenbord. Bij een nieuwe toetsenbordindeling hoort immers een tijd waarin men moet wennen aan de locatie van de letters. Het is daarom belangrijk om de kosten-baten duidelijk te krijgen en ook duidelijk te kunnen maken aan eventuele klanten. Hierbij is een groot voordeel van deze toetsenborden, dat de toetsen breder zijn dan in een QWERTY-toetsenbord en dat mensen op deze manier minder snel de verkeerde toets indrukken. QWERTY heeft

immers op de bovenste rij van het toetsenbord minstens 10 toetsen, terwijl deze toetsenborden er slechts 7 hebben.

Een tweede overweging is de markt zelf. Omdat deze toetsenborden voor het Nederlands geöptimaliseerd zijn, is de markt relatief klein. Daarnaast zorgt de verengelsing voor meer en meer mensen die het Engels verkiezen boven het Nederlands en die dus niet aannemelijk zijn om een Nederlands toetsenbord te gebruiken.

Een derde overweging is hoe het toetsenbord afgemaakt kan worden. Het Toetsenbord Layout Probleem is immers voor slechts 27 karakters uitgevoerd. Nu is het gebruikelijk op de mobiele telefoon om 2 of 3 toetsenborden te hebben, die opgeroepen kunnen worden door een toets, waarin ook leestekens en cijfers en dergelijke te vinden zijn. In figuur 4 en 5 ziet u al 6 extra, nog niet ingevulde, toetsen. Een advies is om die zes in te vullen met in ieder geval de toets die het toetsenbord kan wisselen naar cijfers en leestekens. De overige vijf zouden met veelgebruikte functies als SHIFT, ENTER, BACKSPACE, "," en "." ingevuld kunnen worden, zoals nu overigens bij de meeste mobiele toetsenborden al gebeurd.

Referenties

[1] Birkhoff, G. (1946). Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucumán Rev. Ser. A*, 5, 147-151.

[2] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 203-204.

[3] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 216-217.

[4] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 225-227.

[5] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 250-252.

[6] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 257-259.

[7] Burkard, R.E., Dell'Amico, M., & Martello, S. (2009) *Assignment Problems, Revised Reprint*. Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 259.

[8] Burkard, R. E., & Rendl, F. (1984). A thermodynamically motivated simulation procedure for combinatorial optimization problems. *European Journal of Operational Research, 17*(2), 169-174.

[9] Conrad, K. (1971). *Das Quadratische Zuweisungsproblem und Zwei seiner Spezialfälle*. Mohr.

[10] David, P. A. (1985). Clio and the Economics of QWERTY. *The American Economic Review*, 75(2), 332-337.

[11] Dell'Amico, M., Díaz, J. C. D., Iori, M., & Montanari, R. (2009). The single-finger keyboard layout problem. *Computers & Operations Research, 36*(11), 3002-3012.

[12] Frieze, A. M., & Yadegar, J. (1983). On the quadratic assignment problem. *Discrete Applied Mathematics, 5*(1), 89-98.

[13] Gunraj, D. N., Drumm-Hewitt, A. M., Dashow, E. M., Upadhyay, S. S. N., & Klin, C. M. (2016). Texting insincerely: The role of the period in text messaging. *Computers in Human Behavior*, 55, 1067-1075.

[14] Hajek, B. (1988). Cooling schedules for optimal annealing. *Mathematics of Operations Research, 13*(2), 311-329.

[15] Hardy, G. H., Littlewood, J. E., & Pólya, G. (1952). *Inequalities*. Cambridge University Press, 153.

[16] Ingber, L. (1993). Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling, 18*(11), 29-57.

[17] Klein, R. W., & Dubes, R. C. (1989). Experiments in projection and clustering by simulated annealing. *Pattern Recognition, 22*(2), 213-220.

[18] Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly, 2*(1-2), 83-97.

[19] MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-computer Interaction, 7*(1), 91-139.

[20] Noyes, J. (1983). The QWERTY keyboard: A review. *International Journal of Man-Machine Studies, 18*(3), 265-281.

[21] Sahni, S., & Gonzalez, T. (1976). P-complete approximation problems. *Journal of the ACM (JACM), 23(3)*, 555-565.

[22] Xia, Y. (2008). Gilmore-Lawler bound of quadratic assignment problem. *Frontiers of Mathematics in China, 3*(1), 109-118.

7 Bijlagen

7.1 Matrices

7.1.1 Matrix A

Zie figuur 16 en 17.

7.1.2 Matrix Asym

Zie figuur 18 en 19.

7.1.3 Matrix B6

Zie figuur 20 en 21.

7.1.4 Matrix B4

Zie figuur 22 en 23.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
A	5522	74	459	542	9	347	1544	559	158	14	571	2448	615	5304	3	558	1	3339	857
B	216	376	3	1	2144	0	0	2	892	2	1	182	2	0	323	0	29	220	13
C	172	0	86	4	239	2	1	1846	356	1	43	58	3	1	275	1	0	21	21
D	3773	6	3	276	4304	1	13	16	1166	58	11	29	8	6	878	10	1	255	149
E	162	1140	524	1406	6336	788	913	269	753	22	1311	4675	902	13234	26	214	2	7213	1315
F	82	9	3	89	411	131	49	7	600	16	12	61	3	4	199	2	0	108	123
G	927	6	2	74	4505	1	107	23	284	28	11	49	48	20	813	0	0	357	106
H	1980	2	2	2	3953	1	1	45	863	53	5	3	137	14	1164	1	0	51	3
I	136	14	614	543	5019	64	954	62	59	4465	3124	949	272	2690	54	97	1	123	2217
J	1513	10	0	385	2506	81	69	4	216	3	881	18	6	1317	284	10	0	6	70
K	812	26	1	15	2068	1	9	20	352	132	566	275	38	90	815	16	0	197	171
L	1285	41	6	379	2918	313	150	3	2346	1	214	1770	52	11	431	63	0	4	519
M	2591	67	1	83	2534	1	13	14	1010	13	4	8	307	54	1401	95	1	6	109
N	1175	45	52	2403	1579	34	1057	49	1777	16	699	189	155	582	1269	20	2	30	871
O	40	142	315	219	2879	573	1029	264	502	25	1371	437	1469	1708	3489	1299	1	2459	183
P	543	6	4	43	1082	7	51	20	170	56	2	363	2	11	116	233	0	582	120
Q	2	0	0	1	3	0	0	0	0	0	0	1	0	0	0	0	0	0	2
R	869	107	20	1172	1639	19	807	79	1192	138	452	301	208	112	860	61	1	164	611
S	194	26	511	98	622	13	29	47	151	108	76	328	81	139	282	525	0	24	352
T	496	53	18	39	3525	15	38	346	941	620	14	33	21	10	940	45	0	459	487
U	34	29	133	269	30	47	157	35	1025	12	729	617	87	345	10	242	0	705	648
V	1752	2	3	13	2349	1	5	2	382	1	1	49	12	3	1569	2	0	296	2
W	1501	4	3	31	3545	1	0	41	1103	9	6	7	0	12	564	1	0	6	54
X	1	0	7	19	40	0	0	3	7	0	0	1	0	0	3	6	0	0	0
Y	3	5	11	1	24	2	0	1	1	1	3	7	8	15	46	15	1	1	20
Z	298	0	0	1	1201	0	0	1	1442	2	6	4	1	21	1067	0	1	0	3
∅	2582	2724	503	6079	5084	759	3683	5218	4676	2700	2226	1767	4850	3607	3526	1151	13	671	2179

Figuur 16: Matrix A, alle rijen, kolommen t/m 'S'

	T	U	V	W	X	Y	Z	∅
A	2833	167	372	37	23	21	18	1573
B	71	107	4	1	0	10	1	577
C	132	46	0	0	0	12	0	21
D	102	465	8	19	3	17	11	2220
E	5724	780	848	415	55	94	665	9805
F	366	16	6	15	0	2	67	859
G	122	19	4	9	0	2	15	2836
H	902	446	3	2	5	4	4	449
I	1201	4	90	2	28	0	67	371
J	45	582	56	7	0	0	30	1360
K	505	265	6	87	0	4	5	5065
L	237	333	58	40	2	82	7	3020
M	170	36	3	19	0	30	0	1132
N	825	577	37	41	9	14	131	13019
O	480	788	445	105	6	12	19	564
P	138	62	12	4	0	3	14	933
Q	2	28	0	0	0	1	0	12
R	680	464	130	116	0	93	62	5794
S	2157	256	19	57	1	18	8	4407
T	269	408	30	125	0	19	72	9358
U	198	450	8	357	13	2	33	788
V	13	23	4	1	0	1	1	36
W	7	4	1	50	0	0	3	126
X	33	1	0	0	10	2	0	31
Y	1	0	1	2	0	5	0	194
Z	2	74	2	53	0	2	21	9
∅	2368	674	3993	4585	34	40	2649	427

Figuur 17: Matrix A, alle rijen, kolommen vanaf 'S'

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	145	315.5	2157.5	85.5	214.5	1235.5	1269.5	147	763.5	691.5	1866.5	1603	3239.5
B	145	0	1.5	3.5	1642	4.5	3	2	453	6	13.5	111.5	34.5	22.5
C	315.5	1.5	0	3.5	381.5	2.5	1.5	924	485	0.5	22	32	2	26.5
D	2157.5	3.5	3.5	0	2855	45	43.5	9	854.5	221.5	13	204	45.5	1204.5
E	85.5	1642	381.5	2855	0	599.5	2709	2111	2886	1264	1689.5	3796.5	1718	7406.5
F	214.5	4.5	2.5	45	599.5	0	25	4	332	48.5	6.5	187	2	19
G	1235.5	3	1.5	43.5	2709	25	0	12	619	48.5	10	99.5	30.5	538.5
H	1269.5	2	924	9	2111	4	12	0	462.5	28.5	12.5	3	75.5	31.5
I	147	453	485	854.5	2886	332	619	462.5	0	2340.5	1738	1647.5	641	2233.5
J	763.5	6	0.5	221.5	1264	48.5	48.5	28.5	2340.5	0	506.5	9.5	9.5	666.5
K	691.5	13.5	22	13	1689.5	6.5	10	12.5	1738	506.5	0	244.5	21	394.5
L	1866.5	111.5	32	204	3796.5	187	99.5	3	1647.5	9.5	244.5	0	30	100
M	1603	34.5	2	45.5	1718	2	30.5	75.5	641	9.5	21	30	0	104.5
N	3239.5	22.5	26.5	1204.5	7406.5	19	538.5	31.5	2233.5	666.5	394.5	100	104.5	0
O	21.5	232.5	295	548.5	1452.5	386	921	714	278	154.5	1093	434	1435	1488.5
P	550.5	3	2.5	26.5	648	4.5	25.5	10.5	133.5	33	9	213	48.5	15.5
Q	1.5	14.5	0	1	2.5	0	0	0	0.5	0	0	0.5	0.5	1
R	2104	163.5	20.5	713.5	4426	63.5	582	65	657.5	72	324.5	152.5	107	71
S	525.5	19.5	266	123.5	968.5	68	67.5	25	1184	89	123.5	423.5	95	505
T	1664.5	62	75	70.5	4624.5	190.5	80	624	1071	332.5	259.5	135	95.5	417.5
U	100.5	68	89.5	367	405	31.5	88	240.5	514.5	297	497	475	61.5	461
V	1062	3	1.5	10.5	1598.5	3.5	4.5	2.5	236	28.5	3.5	53.5	7.5	20
W	769	2.5	1.5	25	1980	8	4.5	21.5	552.5	8	46.5	23.5	9.5	26.5
X	12	0	3.5	11	47.5	0	0	4	17.5	0	0	1.5	0	4.5
Y	12	7.5	11.5	9	59	2	1	2.5	0.5	0.5	3.5	44.5	19	14.5
Z	158	0.5	0	6	933	33.5	7.5	2.5	754.5	16	5.5	5.5	0.5	76
∅	2077.5	1650.5	262	4149.5	7444.5	809	3259.5	2833.5	2523.5	2030	3645.5	2393.5	2991	8313

Figuur 18: Matrix A_{sym}, alle rijen, kolommen t/m 'N'

	O	P	Q	R	S	T	U	V	W	X	Y	Z	∅
A	21.5	550.5	1.5	2104	525.5	1664.5	100.5	1062	769	12	12	158	2077.5
B	232.5	3	14.5	163.5	19.5	62	68	3	2.5	0	7.5	0.5	1650.5
C	295	2.5	0	20.5	266	75	89.5	1.5	1.5	3.5	11.5	0	262
D	548.5	26.5	1	713.5	123.5	70.5	367	10.5	25	11	9	6	4149.5
E	1452.5	648	2.5	4426	968.5	4624.5	405	1598.5	1980	47.5	59	933	7444.5
F	386	4.5	0	63.5	68	190.5	31.5	3.5	8	0	2	33.5	809
G	921	25.5	0	582	67.5	80	88	4.5	4.5	0	1	7.5	3259.5
H	714	10.5	0	65	25	624	240.5	2.5	21.5	4	2.5	2.5	2833.5
I	278	133.5	0.5	657.5	1184	1071	514.5	236	552.5	17.5	0.5	754.5	2523.5
J	154.5	33	0	72	89	332.5	297	28.5	8	0	0.5	16	2030
K	1093	9	0	324.5	123.5	259.5	497	3.5	46.5	0	3.5	5.5	3645.5
L	434	213	0.5	152.5	423.5	135	475	53.5	23.5	1.5	44.5	5.5	2393.5
M	1435	48.5	0.5	107	95	95.5	61.5	7.5	9.5	0	19	0.5	2991
N	1488.5	15.5	1	71	505	417.5	461	20	26.5	4.5	14.5	76	8313
O	0	707.5	0.5	1659.5	232.5	710	399	1007	334.5	4.5	29	543	2045
P	707.5	0	0	321.5	322.5	91.5	152	7	2.5	3	9	7	1042
Q	0.5	0	0	0.5	1	1	14	0	0	0	1	0.5	12.5
R	1659.5	321.5	0.5	0	317.5	569.5	584.5	213	61	0	47	31	3232.5
S	232.5	322.5	1	317.5	0	1322	452	10.5	55.5	0.5	19	5.5	3293
T	710	91.5	1	569.5	1322	0	303	21.5	66	16.5	10	37	5863
U	399	152	14	584.5	452	303	0	15.5	180.5	7	1	53.5	731
V	1007	7	0	213	10.5	21.5	15.5	0	1	0	1	1.5	2014.5
W	334.5	2.5	0	61	55.5	66	180.5	1	0	0	1	28	2355.5
X	4.5	3	0	0	0.5	16.5	7	0	0	0	1	0	32.5
Y	29	9	1	47	19	10	1	1	1	1	0	1	117
Z	543	7	0.5	31	5.5	37	53.5	1.5	28	0	1	0	1329
∅	2045	1042	12.5	3232.5	3293	5863	731	2014.5	2355.5	32.5	117	1329	0

Figuur 19: Matrix A_{sym}, alle rijen, kolommen vanaf 'O'

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0.27046	0.32346	0.36885
B	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.32346	0.27046	0.32346
C	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.36885	0.32346	0.27046
D	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40901	0.36885	0.32346
E	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47244	0.40901	0.36885
F	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309	0.5269	0.20408	0.27046	0.36885
G	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046
H	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408
I	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408
J	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046
K	0.50227	0.44284	0.36885	0.27046	0.20408	0.5269	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885
L	0.27046	0.32346	0.36885	0.40901	0.47244	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346
M	0.32346	0.27046	0.32346	0.36885	0.40901	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408
N	0.36885	0.32346	0.27046	0.32346	0.36885	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0
O	0.40901	0.36885	0.32346	0.27046	0.32346	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408
P	0.47244	0.40901	0.36885	0.32346	0.27046	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346
Q	0.36885	0.40719	0.44284	0.47244	0.50285	0.27046	0.32346	0.36885	0.40901	0.47244	0.52754	0.20408	0.27046	0.36885
R	0.36885	0.36885	0.40719	0.44284	0.47244	0.32346	0.27046	0.32346	0.36885	0.40901	0.47244	0.20408	0.20408	0.27046
S	0.40719	0.36885	0.36885	0.40719	0.44284	0.36885	0.32346	0.27046	0.32346	0.36885	0.40901	0.27046	0.20408	0.20408
T	0.44284	0.40719	0.36885	0.36885	0.40719	0.40901	0.36885	0.32346	0.27046	0.32346	0.36885	0.36885	0.27046	0.20408
U	0.47244	0.44284	0.40719	0.36885	0.36885	0.47244	0.40901	0.36885	0.32346	0.27046	0.32346	0.44284	0.36885	0.27046
V	0.50285	0.47244	0.44284	0.40719	0.36885	0.52754	0.47244	0.40901	0.36885	0.32346	0.27046	0.50227	0.44284	0.36885
W	0.40901	0.44284	0.47309	0.50227	0.52754	0.36885	0.36885	0.40719	0.44284	0.47244	0.50285	0.27046	0.32346	0.36885
X	0.44284	0.40901	0.44284	0.47309	0.50227	0.40719	0.36885	0.36885	0.40719	0.44284	0.47244	0.32346	0.27046	0.32346
Y	0.47309	0.44284	0.40901	0.44284	0.47309	0.44284	0.40719	0.36885	0.36885	0.40719	0.44284	0.36885	0.32346	0.27046
Z	0.50227	0.47309	0.44284	0.40901	0.44284	0.47244	0.44284	0.40719	0.36885	0.36885	0.40719	0.40901	0.36885	0.32346
∅	0.52754	0.50227	0.47309	0.44284	0.40901	0.50285	0.47244	0.44284	0.40719	0.36885	0.36885	0.47244	0.40901	0.36885

Figuur 20: Matrix $B6$, alle rijen, kolommen t/m 'N'

	O	P	Q	R	S	T	U	V	W	X	Y	Z	∅
A	0.40901	0.47244	0.36885	0.36885	0.40719	0.44284	0.47244	0.50285	0.40901	0.44284	0.47309	0.50227	0.52754
B	0.36885	0.40901	0.40719	0.36885	0.36885	0.40719	0.44284	0.47244	0.44284	0.40901	0.44284	0.47309	0.50227
C	0.32346	0.36885	0.44284	0.40719	0.36885	0.36885	0.40719	0.44284	0.47309	0.44284	0.40901	0.44284	0.47309
D	0.27046	0.32346	0.47244	0.44284	0.40719	0.36885	0.36885	0.40719	0.50227	0.47309	0.44284	0.40901	0.44284
E	0.32346	0.27046	0.50285	0.47244	0.44284	0.40719	0.36885	0.36885	0.52754	0.50227	0.47309	0.44284	0.40901
F	0.44284	0.50227	0.27046	0.32346	0.36885	0.40901	0.47244	0.52754	0.36885	0.40719	0.44284	0.47244	0.50285
G	0.36885	0.44284	0.32346	0.27046	0.32346	0.36885	0.40901	0.47244	0.36885	0.36885	0.40719	0.44284	0.47244
H	0.27046	0.36885	0.36885	0.32346	0.27046	0.32346	0.36885	0.40901	0.40719	0.36885	0.36885	0.40719	0.44284
I	0.20408	0.27046	0.40901	0.36885	0.32346	0.27046	0.32346	0.36885	0.44284	0.40719	0.36885	0.36885	0.40719
J	0.20408	0.20408	0.47244	0.40901	0.36885	0.32346	0.27046	0.32346	0.47244	0.44284	0.40719	0.36885	0.36885
K	0.27046	0.20408	0.52754	0.47244	0.40901	0.36885	0.32346	0.27046	0.50285	0.47244	0.44284	0.40719	0.36885
L	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0.27046	0.32346	0.36885	0.40901	0.47244
M	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.32346	0.27046	0.32346	0.36885	0.40901
N	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.36885	0.32346	0.27046	0.32346	0.36885
O	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40901	0.36885	0.32346	0.27046	0.32346
P	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47244	0.40901	0.36885	0.32346	0.27046
Q	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309	0.5269	0.20408	0.27046	0.36885	0.44284	0.50227
R	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284
S	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885
T	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046
U	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408
V	0.27046	0.20408	0.5269	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408
W	0.40901	0.47244	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309
X	0.36885	0.40901	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719
Y	0.32346	0.36885	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346
Z	0.27046	0.32346	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408
∅	0.32346	0.27046	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0

Figuur 21: Matrix $B6$, alle rijen, kolommen vanaf 'O'

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
A	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0.32197	0.32346	0.32346
B	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.32346	0.32197	0.32346
C	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.32346	0.32197
D	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40901	0.32346	0.32346
E	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47244	0.40901	0.32346
F	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309	0.5269	0.20408	0.27046	0.36885
G	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046
H	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408
I	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408
J	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046
K	0.50227	0.44284	0.36885	0.27046	0.20408	0.5269	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885
L	0.32197	0.32346	0.32346	0.40901	0.47244	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346
M	0.32346	0.32197	0.32346	0.32346	0.40901	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408
N	0.32346	0.32346	0.32197	0.32346	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0
O	0.40901	0.32346	0.32346	0.32197	0.32346	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408
P	0.47244	0.40901	0.32346	0.32346	0.32197	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346
Q	0.40706	0.40719	0.40816	0.44354	0.50285	0.32197	0.32346	0.32346	0.40901	0.47244	0.52754	0.20408	0.27046	0.36885
T	0.40706	0.40706	0.40719	0.40816	0.44354	0.32346	0.32197	0.32346	0.32346	0.40901	0.47244	0.20408	0.20408	0.27046
S	0.40719	0.40706	0.40706	0.40719	0.40816	0.32346	0.32346	0.32197	0.32346	0.32346	0.40901	0.27046	0.20408	0.20408
T	0.40816	0.40719	0.40706	0.40706	0.40719	0.40901	0.32346	0.32346	0.32197	0.32346	0.32346	0.36885	0.27046	0.20408
U	0.44354	0.40816	0.40719	0.40706	0.40706	0.47244	0.40901	0.32346	0.32346	0.32197	0.32346	0.44284	0.36885	0.27046
V	0.50285	0.44354	0.40816	0.40719	0.40706	0.52754	0.47244	0.40901	0.32346	0.32346	0.32197	0.50227	0.44284	0.36885
W	0.47207	0.47473	0.47309	0.47319	0.47386	0.40706	0.40706	0.40719	0.40816	0.44354	0.50285	0.32197	0.32346	0.32346
X	0.47473	0.47207	0.47473	0.47309	0.47319	0.40719	0.40706	0.40706	0.40719	0.40816	0.44354	0.32346	0.32197	0.32346
Y	0.47309	0.47473	0.47207	0.47473	0.47309	0.40816	0.40719	0.40706	0.40706	0.40719	0.40816	0.32346	0.32346	0.32197
Z	0.47319	0.47309	0.47473	0.47207	0.47473	0.44354	0.40816	0.40719	0.40706	0.40706	0.40719	0.40901	0.32346	0.32346
∅	0.47386	0.47319	0.47309	0.47473	0.47207	0.50285	0.44354	0.40816	0.40719	0.40706	0.40706	0.47244	0.40901	0.32346

Figuur 22: Matrix $B4$, alle rijen, kolommen t/m 'N'

	O	P	Q	R	S	T	U	V	W	X	Y	Z	∅
A	0.40901	0.47244	0.40706	0.40706	0.40719	0.40816	0.44354	0.50285	0.47207	0.47473	0.47309	0.47319	0.47386
B	0.32346	0.40901	0.40719	0.40706	0.40706	0.40719	0.40816	0.44354	0.47473	0.47207	0.47473	0.47309	0.47319
C	0.32346	0.32346	0.40816	0.40719	0.40706	0.40706	0.40719	0.40816	0.47309	0.47473	0.47207	0.47473	0.47309
D	0.32197	0.32346	0.44354	0.40816	0.40719	0.40706	0.40706	0.40719	0.47319	0.47309	0.47473	0.47207	0.47473
E	0.32346	0.32197	0.50285	0.44354	0.40816	0.40719	0.40706	0.40706	0.47386	0.47319	0.47309	0.47473	0.47207
F	0.44284	0.50227	0.32197	0.32346	0.32346	0.40901	0.47244	0.52754	0.40706	0.40719	0.40816	0.44354	0.50285
G	0.36885	0.44284	0.32346	0.32197	0.32346	0.32346	0.40901	0.47244	0.40706	0.40706	0.40719	0.40816	0.44354
H	0.27046	0.36885	0.32346	0.32346	0.32197	0.32346	0.32346	0.40901	0.40719	0.40706	0.40706	0.40719	0.40816
I	0.20408	0.27046	0.40901	0.32346	0.32346	0.32197	0.32346	0.32346	0.40816	0.40719	0.40706	0.40706	0.40719
J	0.20408	0.20408	0.47244	0.40901	0.32346	0.32346	0.32197	0.32346	0.44354	0.40816	0.40719	0.40706	0.40706
K	0.27046	0.20408	0.52754	0.47244	0.40901	0.32346	0.32346	0.32197	0.50285	0.44354	0.40816	0.40719	0.40706
L	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0.32197	0.32346	0.32346	0.40901	0.47244
M	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.32346	0.32197	0.32346	0.32346	0.40901
N	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.32346	0.32197	0.32346	0.32346
O	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40901	0.32346	0.32346	0.32197	0.32346
P	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47244	0.40901	0.32346	0.32346	0.32197
Q	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309	0.5269	0.20408	0.27046	0.36885	0.44284	0.50227
R	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719	0.47309	0.20408	0.20408	0.27046	0.36885	0.44284
S	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346	0.40719	0.27046	0.20408	0.20408	0.27046	0.36885
T	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046
U	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0	0.20408	0.44284	0.36885	0.27046	0.20408	0.20408
V	0.27046	0.20408	0.5269	0.47309	0.40719	0.32346	0.20408	0	0.50227	0.44284	0.36885	0.27046	0.20408
W	0.40901	0.47244	0.20408	0.20408	0.27046	0.36885	0.44284	0.50227	0	0.20408	0.32346	0.40719	0.47309
X	0.32346	0.40901	0.27046	0.20408	0.20408	0.27046	0.36885	0.44284	0.20408	0	0.20408	0.32346	0.40719
Y	0.32346	0.32346	0.36885	0.27046	0.20408	0.20408	0.27046	0.36885	0.32346	0.20408	0	0.20408	0.32346
Z	0.32197	0.32346	0.44284	0.36885	0.27046	0.20408	0.20408	0.27046	0.40719	0.32346	0.20408	0	0.20408
∅	0.32346	0.32197	0.50227	0.44284	0.36885	0.27046	0.20408	0.20408	0.47309	0.40719	0.32346	0.20408	0

Figuur 23: Matrix $B4$, alle rijen, kolommen vanaf 'O'

7.2 Matlabfuncties

7.2.1 Functie movement time

```
function [ MT ] = MT( D,A )
MT=10/49*log2(D/A+1);
end
```

7.2.2 Functie kosten

```
function[z] = kosten(phi,A,B)
z=0;
n=length(A);
for i=1:n
    for j=1:n
        z=z+A(i,j)*B(phi(i),phi(j));
    end
end
end
```

7.2.3 Functie bipartite matching

bron: https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/24134/versions/1/previews/gaimc/bipartite_matching.m/index.html

7.3 Matlabcodes

7.3.1 Code eigenvalue bound

```
A=Asym;
B=B6;

eigvalbound=dot(sort(eig(A)),sort(eig(B),'descend'))
```

7.3.2 Code Gilmore-Lawler bound

```
A=Asym;
B=B6;

n=length(A);
for i=1:n
    a(i,:)=A(i,[1:i-1,i+1:n]);
    b(i,:)=B(i,[1:i-1,i+1:n]);
end
for m=1:length(a)
    for n=1:length(b)
        sortinproduct(m,n)=dot(sort(a(m,:)),sort(b(n,:),'descend'));
    end
end
L=sortinproduct;

%bipartiete matching, deze is voor maximum, dus dat vraagt wat aanpassing
[r,p,phi]=bipartite_matching(max(max(L))+1-L);
z=0;
for i=1:length(L)
    z=z+L(p(i),phi(i));
end
phi
z
```

7.3.3 Code reduced Gilmore-Lawler bound

```
A=Asym;
B=B6;

n=length(A);
r=zeros(1,n);
s=zeros(1,n);
Astreep=zeros(n,n);
Bstreep=zeros(n,n);
for i=1:n
    k=[1:i-1,i+1:n];
    r(i)=min(A(k,i));
    s(i)=min(B(k,i));
    Astreep(k,i)=A(k,i)-r(i);
    Bstreep(k,i)=B(k,i)-s(i);
end
Cstreep=zeros(n,n);
for i=1:n
    for j=1:n
        for m=1:n
            Cstreep(i,j)=Cstreep(i,j)+r(i)*B(m,j)+s(j)*A(m,i);
        end
        Cstreep(i,j)=Cstreep(i,j)-(n-1)*r(i)*s(j);
    end
end
a=zeros(n,n-1);
b=zeros(n,n-1);
for i=1:n
    a(i,:)=Astreep(i,[1:i-1,i+1:n]);
    b(i,:)=Bstreep(i,[1:i-1,i+1:n]);
end
sortinproduct=zeros(n,n);
for m=1:length(a)
    for n=1:length(b)
        sortinproduct(m,n)=dot(sort(a(m,:)),sort(b(n,:),'descend'));
    end
end
L=sortinproduct+Cstreep;

%bipartiete matching, deze is voor maximum, dus dat vraagt wat aanpassing
[r,p,phi]=bipartite_matching(max(max(L))+1-L);
z=0;
for i=1:length(L)
    z=z+L(p(i),phi(i));
end
phi
z
```

7.3.4 Code mean objective value

```
A=Asym;
B=B6;

n=length(A);
```

```
som1=0;
som2=0;
for i=1:n
    for j=1:n
        for k=1:n
            for l=1:n
                som1=som1+A(i,j)*B(k,l);
            end
        end
        som2=som2+A(i,i)*B(j,j);
    end
end
mu=som1/(n*(n-1))+som2/n
```

7.3.5 Code random local search

```
A=Asym;
B=B6;
k=2;

n=length(A);
phi=[1:n];
perm=perms([1:k]);
for t=1:10000
    s=randperm(n,k);
    phiv=phi;
    for m=1:factorial(k)
        phix=phi;
        phix(s(perm(m,k)))=phi(s(perm(m,1)));
        for i=1:k-1
            phix(s(perm(m,i)))=phi(s(perm(m,i+1)));
        end
        if kosten(phix,A,B)<kosten(phiv,A,B)
            phiv=phix;
        end
    end
    phi=phiv;
end
phi
z=kosten(phi,A,B)
```

7.3.6 Code gestructureerde local search

Voor $k = 2$:

```
A=Asym;
B=B6;
phi=[1:27];

n=length(A);
change=1;
while change==1
    change=0;
    phiv=phi;
    for i=1:n
```

```
        for j=1:n
            phix=phi;
            phix(i)=phi(j);
            phix(j)=phi(i);
            if kosten(phix,A,B)<kosten(phiv,A,B)
                phiv=phix;
                change=1;
            end
        end
    end
    phi=phiv;
end
phi
z=kosten(phi,A,B)
```

Voor $k = 3$:

```
A=Asym;
B=B6;
phi=[1:27];

k=3;
n=length(A);
change=1;
while change==1
    change=0;
    phiv=phi;
    for h=1:n-2
        for i=h+1:n-1
            for j=i+1:n
                perm=perms([h,i,j]);
                for m=1:factorial(k)
                    phix=phi;
                    phix(h)=phi(perm(m,1));
                    phix(i)=phi(perm(m,2));
                    phix(j)=phi(perm(m,3));
                    if kosten(phix,A,B)<kosten(phiv,A,B)
                        phiv=phix;
                        change=1;
                    end
                end
            end
        end
    end
    phi=phiv;
end
phi
z=kosten(phi,A,B)
```

7.3.7 Code random simulated annealing

```
A=Asym;
B=B6;
k=2;
T=5000;

n=length(A);
phi=[1:n];
perm=perms([1:k]);
for t=1:10000
    s=randperm(n,k);
    phiv=phi;
    for m=1:factorial(k)
        phix=phi;
        phix(s(perm(m,k)))=phi(s(perm(m,1)));
        for i=1:k-1
            phix(s(perm(m,i)))=phi(s(perm(m,i+1)));
        end
        if kosten(phix,A,B)<kosten(phiv,A,B)
            phiv=phix;
        elseif exp((kosten(phiv,A,B)-kosten(phix,A,B))/T) > rand(1)
            phiv=phix;
        end
    end
    if rem(t,2*n)==0
        T=T/2;
    end
    phi=phiv;
end
phi
z=kosten(phi,A,B)
```

7.3.8 Code gestructureerde simulated annealing

```
n=27;
A=Asym;
B=B6;
T=5000;

phi=[1:n];
for t=1:10000
    for i=1:n
        for j=1:n
            phix=phi;
            phix(i)=phi(j);
            phix(j)=phi(i);
            if kosten(phix,A,B)<kosten(phi,A,B)
                phi=phix;
            elseif exp((kosten(phi,A,B)-kosten(phix,A,B))/T) > rand(1)
                phi=phix;
            end
        end
    end
    if rem(t,2*n)==0;
        T=T/2;
    end
end
phi
z=kosten(phi,A,B)
```

7.3.9 Code herhaaldelijke gestructureerde simulated annealing

```
A=Asym; %deze is dus voor k=2
B=B6;

n=length(A);
phi=[1:n];
while kosten(phi,A,B)>101500
T=5000;
t=0;
phi=randperm(n,n);
change=1;
while change==1
    change=0;
    for i=1:n-1
        for j=i+1:n
            phix=phi;
            phix(i)=phi(j);
            phix(j)=phi(i);
            if kosten(phix,A,B)<kosten(phi,A,B)
                phi=phix;
                change=1;
            elseif exp((kosten(phi,A,B)-kosten(phix,A,B))/T) > rand(1)
                phi=phix;
                change=1;
            end
            t=t+1;
            if rem(t,2*n)==0
                T=T/2;
            end
        end
    end
end
end
phi
kosten(phi,A,B)
```