

Het handboek van Rocs

Tomaz Canabrava
Andreas Cord-Landwehr
Vertaler/Nalezer: Freek de Kruijf
Vertaler/Nalezer: Jaap Woldringh



Het handboek van Rocs

Inhoudsopgave

1	Inleiding	6
1.1	Doelen, voor wie is dit bestemd en manier van werken.	6
1.2	Rocs in een notendop	7
1.2.1	Graafdocumenten	7
1.2.2	Kanttypen	7
1.2.3	Knooptypen	7
1.2.4	Eigenschappen	8
1.3	Instructie	8
1.3.1	Een graaf genereren	8
1.3.2	De elementtypen genereren	8
1.3.3	Het algoritme	9
1.3.4	Het algoritme uitvoeren	9
2	De gebruikersinterface van Rocs	10
2.1	Belangrijkste elementen van de gebruikersinterface	10
3	Scripts maken	12
3.1	Het uitvoeren van algoritmes in Rocs	12
3.1.1	Sturen uitvoeren van script	12
3.1.2	Scriptuitvoer	12
3.1.3	API voor scripts	12
4	Import en export	14
4.1	Rocs-projecten uitwisselen	14
4.1.1	Import en export van graafdocumenten	14
4.1.1.1	Trivial Graph bestandsformaat	14
4.1.1.1.1	Formaatbeschrijving	14
4.1.1.1.2	Voorbeeld	15
4.1.1.2	DOT-taal / Graphviz Graph bestandsformaat	15
4.1.1.2.1	Niet ondersteunde eigenschappen	15
4.1.1.2.2	Voorbeeld	15

5 Graafindeling	16
5.1 Automatische indeling van grafen in Rocs	16
5.1.1 Op krachten gebaseerde Indeling	16
5.1.1.1 Radiële boom	17
6 Dankbetuiging en licentie	18

Samenvatting

Rocs is een hulpmiddel voor de grafentheorie.

Hoofdstuk 1

Inleiding

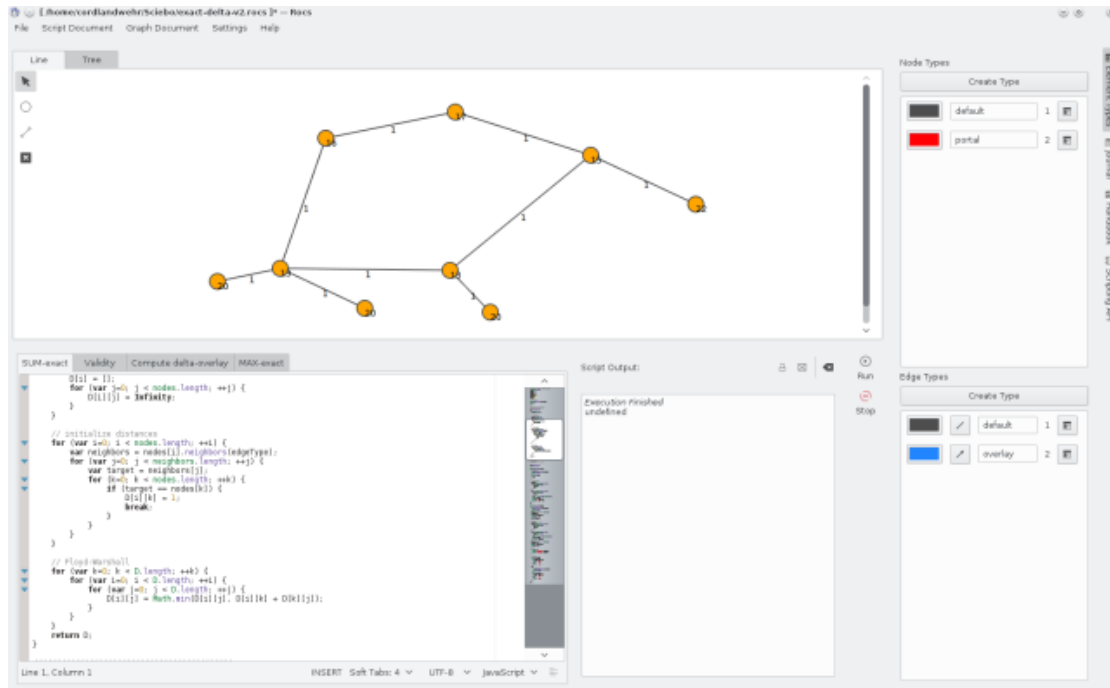
In dit hoofdstuk vindt u een overzicht van de kerneigenschappen en manier van werken met Rocs. Het belangrijkste zijn Section 1.2 en hoofdstuk 3, hiermee kan elke nieuwe gebruiker onmiddellijk aan de slag gaan met Rocs.

1.1 Doelen, voor wie is dit bestemd en manier van werken.

Rocs is een hulpmiddel voor de grafentheorie, geschikt voor iedereen die algoritmes voor grafen wil ontwerpen en bestuderen. Dit zijn vooral

- docenten die algoritmes aan hun studenten willen demonstreren,
- studenten en onderzoekers die willen begrijpen en zien hoe een algoritme werkt, en
- iedereen die belang stelt in gegevensstructuren en algoritmes.

Voor ieder van hen heeft Rocs een eenvoudig grafisch bewerkingsprogramma voor het aanmaken van grafen, een krachtig hulpmiddel voor het maken van scripts voor algoritmen, en diverse hulpmiddelen voor simulaties, experimenten, en het exporteren van grafen. Rocs kan typisch worden gebruikt voor het aanmaken van een graaf, handmatig (met slepen en neerzetten van knopen en kanten op het whiteboard), of met een van de graafgeneratoren. Op de aangemaakte graaf kunnen daarna algoritmen worden aangebracht en toegepast, en alle veranderingen als gevolg hiervan zijn direct zichtbaar in de graafbewerker.



1.2 Rocs in een notendop

Elke sessie in Rocs is een project: wanneer Rocs wordt gestart, wordt een leeg project aangemaakt. Bij het inlezen of importeren van een ander project wordt dat het huidige project. Hierbij bestaat een project zelf uit *graafdocumenten*, *scripts/algoritmen*, en een *journaal*.

1.2.1 Graafdocumenten

Een graafdocument bevat de inhoud van een whiteboard, in de graafbewerker. Hierin is informatie over de door de gebruiker gedefinieerde knoop- en kanttypen, samen met hun eigenschappen, en de reeds aangemaakte knopen en kanten. Dit betekent dat Rocs de verzameling kent van alle knopen en kanten in een graafdocument, en er een (niet noodzakelijk verbonden) graaf mee kan vormen. Alles dat tot een graafdocument behoort, is toegankelijk voor het programma waarmee scripts worden gemaakt, via het globale object **Document**.

1.2.2 Kanttypen


In sommige scenario's hebben grafen verschillende soorten kanten (bijv. een ongerichte graaf plus de kanten in een boom berekend met een "breadth-first" zoekalgoritme) die anders worden behandeld en weergegeven. U kunt hiervoor zelf, naast een standaard kanttype, andere kanttypen definiëren. Elk kanttype wordt op een eigen manier visueel weergegeven, heeft eigen dynamische eigenschappen, en kan gericht of ongericht worden gemaakt. De interface voor scripts bevat mogelijkheden om eenvoudig de toegang te regelen voor alleen bepaalde kanttypen.

1.2.3 Knooptypen

Op dezelfde manier als bij kanttypen, kunnen verschillende knooptypen voor een graaf worden gedefinieerd (bijv. om sommige knopen een speciale rol te geven). Elk knooptype wordt op een eigen manier visueel weergegeven, en heeft eigen dynamische eigenschappen.

1.2.4 Eigenschappen

Elk element (knoop of kant) kan eigenschappen hebben. Deze eigenschappen moeten worden ingesteld bij het overeenkomende knoop- of kanttype. Eigenschappen hebben een naam, en kunnen elke waarde bevatten. Het nieuw aanmaken of wijzigen van bestaande eigenschappen, kunt

u in de zijbalk **Elementtypen** doen met de knop  **Eigenschappen**. U ziet dan een eigenschappendialoog.

U kunt ook met het scriptprogramma bestaande eigenschappen vinden en de waarden hiervan wijzigen. In het volgende voorbeeld nemen we aan dat de eigenschap 'gewicht' bestaat voor het standaard kanttype.

```
var knopen = Document.nodes()
for (var i = 0; i < knopen.length; ++i){
    knopen[i].gewicht = i;
}
for (var i = 0; i < knopen.length; ++i){
    Console.log("gewicht knoop " + i + ": " + knopen[i].gewicht);
}
```

1.3 Instructie


In dit gedeelte maken we een voorbeeldproject aan, waarin enkele van de meest belangrijke eigenschappen van Rocs worden gebruikt. Het doel is een graaf en een script te maken voor een eenvoudig 2-benaderingsalgoritme voor het *“minimum vertex cover”*-probleem. Dit houdt in het zoeken naar de kleinste deelverzameling C van de knopen van de graaf, zodat elke kant van de graaf met tenminste een knoop van C is verbonden. Het is bekend dat dit probleem “NP-hard” is, en we willen laten zien hoe we een benadering met een factor 2 kunnen vinden door een overeenkomst in de bestaande graaf te zoeken (in het volgende gebruiken we de gebruikelijke termen in graafalgoritmes: graaf is de gegevensstructuur, knopen zijn de gegevenselementen, kanten zijn de pijlen). (Noot vertaler: ik handhaaf vele Engelse benamingen, ten eerste om de gebruiker van deze tekst in de gelegenheid te stellen om naar deze namen te googlen, en ten tweede omdat ik van de meeste niet de Nederlandse naam (als die er is) ken, of kan vinden).

Ons doel is de overeenkomst zichtbaar te maken tussen de overeenkomende en de *“minimum vertex cover”*. Hiervoor moeten we twee kanttypen specificeren, een voor het tonen van de overeenkomende kanten, en een voor de ‘gewone’ kanten. En verder twee knooptypen, die gebruikt worden voor het onderscheid tussen knopen in C en knopen die niet tot C behoren.

1.3.1 Een graaf genereren

Voor het aanmaken van de graaf is in Rocs een hulpmiddel aanwezig waarmee grafen kunnen worden aangemaakt. We gaan naar het menu **Graafdocument** → **Hulpmiddelen** → **Graaf aanmaken**. Hier genereren we een **Willekeurige graaf**, met 30 knopen, 90 kanten en met het zaadje 1 (het zaadje is de startwaarde voor het genereren van willekeurige getallen; meerdere keren dit zelfde zaadje gebruiken resulteert in dezelfde, en dus reproduceerbare grafen).

1.3.2 De elementtypen genereren

Met **Elementtypen** maken we een tweede knooptype aan en een tweede kanttype. Voor beide nieuwe typen openen we de eigenschappendialoog met respectievelijk de knoppen  **Eigenschappen** en zetten de ID's op 2. Verder wijzigen we de kleuren van elementen van de beide nieuwe typen (om ze van de standaardtypen te onderscheiden). Tenslotte maken we alle kanttypen bidirectioneel en de ID's van de standaardtypen 1.

1.3.3 Het algoritme

Tenslotte moeten we het benaderingsalgoritme implementeren. Hiervoor gebruiken we het volgende:

```

for (var i=0; i < Document.nodes.length; i++) {
    Document.nodes[i].type = 1;
}
for (var i=0; i < Document.edges.length; i++) {
    Document.edges[i].type = 1;
}

var E = Document.edges(); // verzameling onverwerkte kanten
var C = new Array();      // overeenkomende kanten
while (E.length
> 0) {
    var e = E[0];          // we nemen eerste kant e={u,v}
    var u = e.from();
    var v = e.to();
    e.type = 2;           // maak kant overeenkomend
    E.shift();            // verwijder e (bijv. E[0]) uit kantenlijst
    C.push(u);            // u aan C toevoegen
    C.push(v);            // v aan C toevoegen

    // u,v als knopen in C markeren
    u.type = 2;
    v.type = 2;

    // uit E alle kanten verwijderen die verbonden zijn met u of v
    var aanliggend = u.edges();
    for (var i=0; i < aanliggend.length; i++) {
        var index = E.indexOf(aanliggend[i]); // de index bepalen
        if (index != -1) {
            E.splice(index, 1); // het verwijderen indien gevonden
        }
    }
    var aanliggend = v.edges();
    for (var i=0; i < aanliggend.length; i++) {
        var index = E.indexOf(adjacent[i]); // de index bepalen
        if (index != -1) {
            E.splice(index, 1); // het verwijderen indien gevonden
        }
    }
}
Console.log("Vertex Cover bevat " + C.length + " knopen.");





```

1.3.4 Het algoritme uitvoeren

Het algoritme kunnen we starten met de knop



Uitvoeren in het controlepaneel voor scripts.

-  **Selecteren en verplaatsen:** U kunt elementen selecteren door in het whiteboard op een lege plek te klikken, of door de muisknop ingedrukt te houden en een rechthoek te “trekken” om enige gegevenselementen heen om deze elementen te selecteren, of door direct op een niet geselecteerd element te klikken om dit te selecteren. Indien u op een geselecteerd element klikt, of op een verzameling van geselecteerde elementen, kunt u die verplaatsen met de muis, met ingedrukte muisknop. Geselecteerde elementen kunnen ook met de pijltjestoetsen worden verplaatst.
-  **Een knoop aanmaken:** Klik op een willekeurige plaats in het whiteboard van het grafische bewerkingsprogramma, om zo een nieuw gegevenselement aan te maken voor de huidige geselecteerde gegevensstructuur. Door de muisaanwijzer op de knop ingedrukt te houden, komt er een contextmenu, waarin het type kan worden gekozen van de nieuw aangemaakte gegevenselementen (alleen wanneer er meerdere gegevenstypen zijn).
-  **Een kant aanmaken:** Klik op een gegevenselement, houdt de muisknop ingedrukt, en trek een lijn naar een ander gegevenselement waarnaar de kant moet wijzen. Dit zal alleen lukken als het in de huidige graaf is toegestaan deze kant toe te voegen (bijv., in een ongerichte graaf kunt u niet meerdere kanten toevoegen tussen gegevenselementen). Door de muisaanwijzer op de knop ingedrukt te houden, komt er een contextmenu, waarin het type kan worden gekozen van de nieuw aangemaakte pijlen (alleen wanneer daarvoor meerdere kanttypen zijn).
-  **Element verwijderen:** Klik op het element om het te verwijderen. Bij het verwijderen van een knoop worden meteen alle ermee verbonden kanten verwijderd.

Zijbalk

Rechts vindt u de zijbalk waarin u diverse hulpmiddelen vindt om mee te werken:

- **Elementtypen:** Hiermee heeft u direct toegang tot de beschikbare kant- en knooptypen.
- **Journal:** Bij elk project hoort een eigen journal waarin bijv. taken, resultaten of waarnemingen kunnen worden genoteerd.
- **API voor scripts:** Hiermee kunt u direct toegang krijgen tot de documentatie over scripts, en zo tot de documentatie over scripts.

Bewerken van scripts

In deze tekstbewerker kunt u algoritmes schrijven, zoals in detail is uitgelegd in hoofdstuk 3. U kunt aan verschillende scriptdocumenten tegelijk werken, in verschillende tabbladen.


Scriptuitvoer:

In dit tekstgebied staat informatie over het herstellen van fouten (debug), of de scriptuitvoer van uw algoritme, afhankelijk van de selectie van de ingestelde uitvoer, bovenin. Als het script een fout meldt, wordt automatisch de debug-uitvoer geselecteerd.

Besturing

Hierin kunt u het uitvoeren van scripts regelen. U kunt het script uitvoeren (draaien, laten

werken) dat op dit moment in de scriptbewerker is geopend, door op de knop  **Uitvoeren** te klikken. Een script kan tijdens het uitvoeren worden gestopt door op de knop

 **Stop** te klikken.

Hoofdstuk 3



Scripts maken

3.1 Het uitvoeren van algoritmes in Rocs

Intern maakt Rocs gebruik van QTScript JavaScript. Dit betekent dat voor alle algoritmen JavaScript nodig is. We leggen u hier nu uit hoe u tot elementen in een graafdocument toegang heeft, en die kunt wijzigen, bij het maken van scripts. Het is belangrijk te weten dat alle wijzigingen in een script direct werken op de eigenschappen van de elementen in de graafbewerker.

3.1.1 Sturen uitvoeren van script

U kunt scripts op verschillende manieren uitvoeren:

-  **Uitvoeren:** Script uitvoeren tot het einde.
-  **Stop:** Stop het uitvoeren van script (alleen wanneer een script bezig is).

3.1.2 Scriptuitvoer

Gedurende het uitvoeren (werking) van een algoritme, worden debug (programmafouten) en programmauitvoer getoond in de *Debug ∧ scriptuitvoer*. Indien er een syntaxfout in uw script wordt ontdekt, wordt de fout ook getoond als een debugbericht. Merk op dat alle programmaberichten ook in de debuguitvoer worden getoond (in een vette letter).

U kunt met de volgende functies de tekst regelen, die in de scriptuitvoer wordt getoond:

```
Console.log(tekenreeks);           // toont tekenreeks als scriptuitvoer
  Console.debug(tekenreeks);       // toont tekenreeks als ↔
    debuguitvoer
  Console.error(tekenreeks);       // toont tekenreeks als foutuitvoer
```

3.1.3 API voor scripts

De diverse onderdelen van Rocs hebben elk een statisch element dat in scripts kan worden gebruikt. Dit zijn:

Het handboek van Rocs

- **Document** voor het graafdocument
- **Console** voor de loguitvoer van de console

Voor het expliciete API-gebruik en voor een methodenoverzicht, zie de help die in de zijbalk van Rocs beschikbaar is.

Hoofdstuk 4

Import en export

4.1 Rocs-projecten uitwisselen

Rocs-projecten kunnen worden geïmporteerd en geëxporteerd als archiefbestanden (.tar.gz-bestanden). Deze archieven kunnen worden gebruikt om projecten te ruilen met anderen. Importeren en exporteren doet u met respectievelijk de menu-items **Graafdocument** → **Graaf importeren** en **Graafdocument** → **Graaf exporteren als**.

4.1.1 Import en export van graafdocumenten

Import en export naar de volgende formaten worden door Rocs ondersteund:

- DOT-bestanden, ook wel bekend als Graphviz-bestanden.
- GML-bestanden
- Trivial Graph Format bestanden
- Keyhole Markup Language formaat

4.1.1.1 Trivial Graph bestandsformaat

Trivial Graph Format (TGF) is een formaat waarin een grafiek in een eenvoudig tekstbestand wordt beschreven. Een TGF-bestand bevat een lijst van knoopdefinities, waarin knoop-ID's worden gekoppeld aan namen, gevolgd door een lijst van kanten. In dit formaat is slechts één naam mogelijk per knoop, en één waarde per kant. Rocs beschouwt geïmporteerde grafen als niet gericht. Geëxporteerde grafen bevatten twee kanten voor elke bi-directionele verbinding (naar beide richtingen).

4.1.1.1.1 Formaatbeschrijving

- Het bestand begint met een lijst van knopen (op elke regel een knoop), gevolgd door een regel met als enige karakter een '#', gevolgd door een lijst van kanten (op elke regel een kant).
- Een knoop wordt beschreven met een geheel getal (als identificatie), gevolgd door een spatie, gevolgd door een willekeurige tekenreeks.
- Een kant wordt beschreven door twee gehele getallen (die de verbonden knopen identificeren), gescheiden door een spatie, gevolgd door een spatie, gevolgd door een willekeurige tekenreeks. Aangenomen wordt dat de gerichte kant wijst van de eerste naar de tweede geïdentificeerde knoop.

4.1.1.1.2 Voorbeeld

```
1 beginknoop
2 zender
3 aarde
#
1 2 blauw
2 1 rood
2 3 groen
```

4.1.1.2 DOT-taal / Graphviz Graph bestandsformaat

De taal DOT is een taal voor het beschrijven van een graaf in klare tekst, waarin een goede voor mensen leesbare beschrijving wordt gegeven van grafen, die ook geschikt is voor het efficiënt verwerken door programma's voor het indelen van grafen. DOT is het standaard bestandsformaat voor de Graphviz programma's voor het visualiseren van grafen, maar wordt ook algemeen in andere graafprogramma's gebruikt. De gebruikelijke bestandsextensies voor DOT zijn `.gv` en `.dot`.

4.1.1.2.1 Niet ondersteunde eigenschappen

Rocs kan elk graafbestand lezen van een graaf die volgens DOT is beschreven ¹. De ondersteuning van deze taal is volledig, ondanks de volgende uitzonderingen:

- subgraaf: Omdat in Rocs het begrip subgraaf niet bestaat, worden subgrafen slechts geïmporteerd als verzamelingen van gegevens-elementen en verbindingen. In het bijzonder worden verbindingen naar en van subgrafen niet geïmporteerd.
- HTML en XML attributen: Attributen (zoals tekst) die HTML of XML bevatten worden onveranderd gelezen. In het bijzonder worden aanpassingen van lettertypen en stijlen van deze attributen niet gelezen.

4.1.1.2.2 Voorbeeld

```
digraph mijnGraaf {
  a -> b -> c;
  b -> d;
}
```

¹<https://graphviz.org/doc/info/lang.html>

Hoofdstuk 5

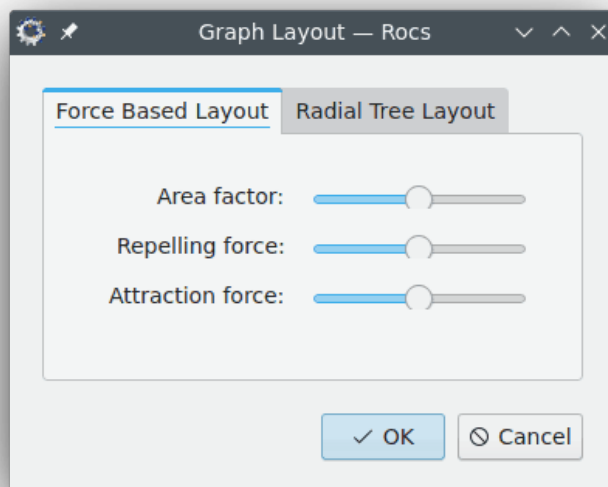
Graafindeling

5.1 Automatische indeling van grafen in Rocs

Rocs kan grafen automatisch indelen. Het hulpmiddel hiervoor is te vinden in het hoofdmenu **Graaf Document** → **Hulpmiddelen** → **Graafindeling**. Er zijn twee verschillende algoritmen die kunnen worden gebruikt: Op krachten gebaseerde indeling, en de radiële boomindeling. U kunt het te gebruiken algoritme in het bijbehorende tabblad te kiezen in het hulpmiddel voor het indelen van grafen. Kies hierin de gewenste parameters, en voer het algoritme uit door te drukken op de knop **OK**. In de volgende secties leest u over de details voor beide indelingsalgoritmen.

5.1.1 Op krachten gebaseerde Indeling

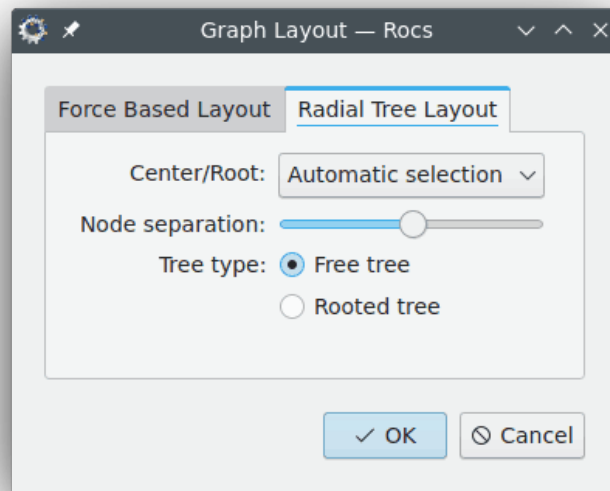
De op krachten gebaseerde indeling kan in elke graaf worden toegepast. Op intuïtieve manier simuleert dit algoritme de krachten die in elk knoop optreden. Er zijn afstotende krachten, en aantrekkende krachten tussen knopenparen die elkaars buurman zijn. Met behulp van schuifknoppen in de gebruikersinterface kan de grootte van deze krachten worden ingesteld.



Nog een parameter die kan worden bestuurd is de Oppervlaktefactor. Deze parameter regelt de afstanden tussen de knopen. Hoge waarden van deze factor leiden tot grotere afstanden tussen de knopen.

5.1.1.1 Radiële boom

De radiële boomindeling kan alleen op bomen worden toegepast. Elke poging die op andere soorten grafen toe te passen geeft een foutmelding. In de aanwezige gebruikersinterface kunnen de parameters worden gekozen voor de radiële boomindeling.



Met de parameter voor boomtype kiest u tussen een vrije boomindeling en een indeling met een wortel. In een losse indeling worden de knopen willekeurig geplaatst zonder een kenbare hiërarchie. In een boom met een wortel wordt de wortel bovenaan geplaatst, en subbomen worden er onder geplaatst, zodat er een hiërarchie ontstaat tussen de knopen.

De parameter voor centrum/wortel bepaalt de knoop die als wortel wordt gebruikt in de indeling als boom of als centrum in de vrije indeling. Het centrum van de vrije indeling is de eerste knoop die door het algoritme wordt geplaatst. Alle andere knopen komen op cirkels rondom de middelste knoop te liggen. Een centrum/wortel kan automatisch worden gekozen door het algoritme voor de indeling.

De parameter voor de afstand tussen de knopen regelt de afstand tussen de knopen. Een grotere waarde maakt die afstand groter. Omgekeerd maakt een kleinere waarde die afstand kleiner.

Hoofdstuk 6

Dankbetuiging en licentie

Rocs

Programma Copyright:

- Copyright 2008 Ugo Sangiori (ugorox AT gmail.com)
- Copyright 2008-2012 Tomaz Canabrava (tcanabrava AT kde.org)
- Copyright 2008-2012 Wagner Reck (wagner.reck AT gmail.com)
- Copyright 2011-2015 Andreas Cord-Landwehr (cordlandwehr AT kde.org)

Documentatie Copyright:

- Documentatie copyright 2009 Anne-Marie Mahfouf annma@kde.org
- Documentatie copyright 2009 Tomaz Canabrava (tcanabrava AT kde.org)
- Documentation copyright 2011-2015 Andreas Cord-Landwehr (cordlandwehr AT kde.org)

Op- of aanmerkingen over de vertalingen van de toepassing en haar documentatie kunt u melden op <http://www.kde.nl/bugs>.

Dit document is vertaald in het Nederlands door Freek de Kruijf freekdekruijf@kde.nl.

Dit document is vertaald in het Nederlands door Jaap Woldringh [op kde punt nl](http://kde.punt.nl).

Deze documentatie valt onder de bepalingen van de [GNU vrije-documentatie-licentie](#).

Deze toepassing valt onder de bepalingen van de [GNU General Public License](#).