Self-Reconfigurable Networks: een Case Study over een Elektriciteitsnetwerk

Sander Borsboom, Niels Monshouwer, Jop van Raaij,
Ton Wessling, Janneke van der Zwaan
olv.
Marinus Maris

Januari 2004, FNWI, UvA

Abstract

In dit verslag wordt een self-reconfigurable network beschreven toegepast op een elektriciteitsnetwerk. Daarnaast worden verschillende beslissingsmethoden besproken en geëvalueerd. Het blijkt dat geen enkele methode op zich voldoende is.

1. Inleiding

Een sensornetwerk kan gebruikt worden om bepaalde systemen in de gaten te houden. Als de sensoren constateren dat er wat mis gaat, is het vaak van groot belang dat er snel gereageerd wordt op de veranderde situatie. Als gevolg van de opgetreden fout, zou het systeem geheel of, in een gunstiger geval, gedeeltelijk uitgevallen kunnen zijn. Pas als door menselijk optreden de fout verholpen is, functioneert het systeem weer naar behoren. Sommige systemen zijn onmisbaar. Men kan het zich dan niet veroorloven dat ze uitvallen. Dit probleem kan opgelost worden door het hele systeem redundant uit te voeren. Dat kost echter veel geld en onderhoud. Een andere oplossing is het systeem intelligent te maken, zodat zelfstandig en autonoom nieuwe instellingen worden gekozen waarmee het probleem omzeild of zelfs opgelost wordt.

Systemen die zichzelf zelfstandig en autonoom opnieuw instellen als er wat mis gaat worden *self-reconfigurable* systemen genoemd. Een *self-reconfigurable* systeem bestaat uit een netwerk van sensoren en actuatoren. Iedere sensor heeft een uniek adres en kan communiceren met de rest van het netwerk (direct of via zijn buren). Het netwerk bestaat uit verschillende soorten knooppunten, zoals specifieke sensoren, knooppunten die (alleen) berichten doorgeven (*routing nodes*), knooppunten waar data samengevoegd wordt (*aggregating nodes*) en knooppunten waar data opgeslagen kan worden (*sink nodes*).

Het netwerk neemt intelligente beslissingen op basis van de verschillende sensorwaardes. Een deel van de intelligentie bevindt zich in de sensors zelf. Slimme sensors bewerken de ruwe data die ze binnen krijgen, voordat het wordt doorgegeven aan het netwerk.

Bovendien kunnen ze zelfstandig en autonoom beslissingen nemen op basis van bepaalde sensorwaardes en bijvoorbeeld opdrachten geven aan actuatoren. Deze lokale acties zijn natuurlijk behoorlijk beperkt. Een gemiddeld systeem bestaat uit vele sensoren en om dan zinvolle beslissingen te kunnen nemen, is de waarde van een afzonderlijke sensor meestal niet voldoende. De sensoren moeten dus samenwerken om een beslissing te kunnen nemen. Om te voorkomen dat het systeem afhankelijk is van één enkele centrale computer, zal dit proces gedistribueerd moeten verlopen. Ook is het handig de sensoren te groeperen, omdat vaak lang niet alle sensors nodig zijn om een bepaald lokaal probleem op te lossen. Als een cluster van sensoren niet in staat is het probleem op te lossen, wordt het doorgegeven aan een hogere intelligente laag. Er wordt dus een onderscheid gemaakt tussen drie niveaus. Ten eerste is er het sensorniveau. Op het sensorniveau worden beslissingen genomen op basis van de sensorwaardes en bijbehorende regels. Op dit niveau is het eigenlijk onmogelijk een intelligente methode te gebruiken die niet gebaseerd is op regels. Op clusterniveau en op globaal niveau is er wel ruimte voor verschillende gedistribueerde methodes om beslissingen te nemen.

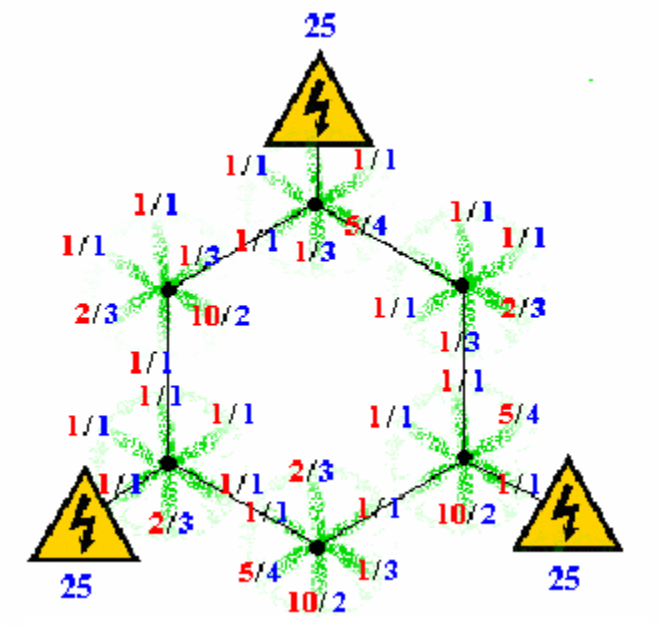
In dit artikel vergelijken we een aantal van die methodes met elkaar aan de hand van een casus. In de volgende paragraaf worden de casus en drie testscenario's uitgewerkt. In paragraaf drie worden een aantal gedistribueerde methodes om beslissingen te maken opgesomd en uitgelegd. In paragraaf vier worden de methodes vergeleken op bepaalde eigenschappen. Ook wordt de relevantie van de eigenschappen besproken voor de testscenario's. De conclusies volgen in paragraaf vijf.

2. De Opbouw van het Netwerk met mogelijke Scenario's

2.1 Opbouw van het netwerk

De casus bestaat uit een elektriciteitsnetwerk. Het netwerk is opgebouwd uit vier typen objecten: Energiecentrales, *Nodes*, Gebruikers en Verbindingen (elektriciteitsdraden). Voor de communicatie is een apart netwerk aanwezig. De elektriciteitsdraden zijn dus niet perse communicatielijnen.
Het elektriciteitsnetwerk is opgebouwd uit clusters met een ringstructuur. Gebruikers zijn eindgebruikers, zoals woonwijken, fabrieken, ziekenhuizen, etc. Elke Gebruiker heeft een Vraag voor stroom (hoeveel energie de Gebruiker nodig heeft) en een Prioriteit. De Prioriteit is een maat voor de belangrijkheid van de Gebruiker, hoe hoger de Prioriteit, hoe meer moeite er gedaan moet worden om stroom naar die Gebruiker te laten gaan. Ziekenhuizen hebben bijvoorbeeld een hogere prioriteit dan woonwijken. Gebruikers krijgen stroom van de *Nodes*. Deze *Nodes* krijgen van hun aangesloten Gebruikers een Vraag met Prioriteit binnen en proberen daar aan te voldoen. *Nodes* kunnen ook een Vraag met Prioriteit krijgen van andere *Nodes*. Niet alle *Nodes* zijn direct gekoppeld aan een stroomvoorziening. Naar gelang de prioriteiten en de hoeveelheid beschikbare stroom, zullen de *Nodes* beslissen hoeveel stroom waar heen gaat. De hogere intelligente laag kan de *Nodes* opleggen hoe de inkomende stroom verdeeld moet worden. Het intelligente hogere netwerk communiceert met alle *Nodes*. Alle stroom in het netwerk wordt geleverd door Energiecentrales. Dit kan een steenkoolcentrale zijn, een nucleaire

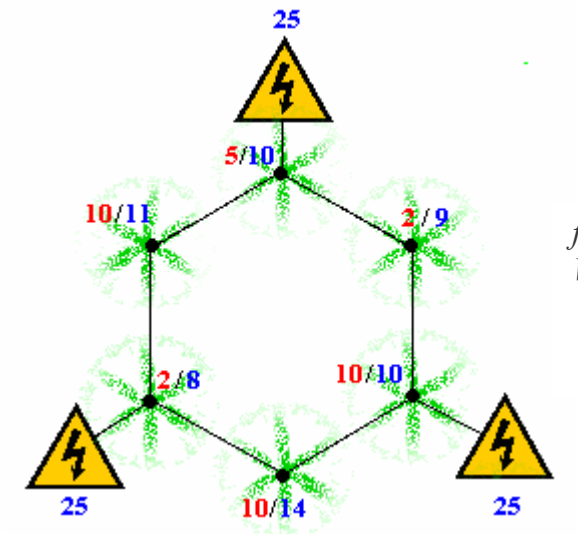
krachtcentrale, windmolens, etc. Deze Energiecentrales kunnen van het hogere netwerk te horen krijgen dat zij meer of minder moeten gaan produceren, en ze kunnen zelf aan het netwerk aangeven dat meer stroom niet mogelijk is.
Het netwerk detecteert ook of verschillende onderdelen kapot zijn. Er komt bijvoorbeeld geen stroom meer binnen of er is geen communicatie meer mogelijk. Als er een probleem gedetecteerd is, moet er reconfiguratie van het netwerk plaatsvinden, en eventueel ook van de Energiecentrales en de Gebruikers.



Figuur 1a: een netwerk met de Gebruikers' Vraag en Priorities zichtbaar

In de afbeeldingen zijn de zwarte stippen de *Nodes*, en de groene sterren zijn de netwerkjes naar de Gebruikers. De rode cijfers geven de Prioriteit aan, de blauwe de Vraag. 1/1 is typisch iets voor een woonwijk; lage Prioriteit en lage Vraag. Bij de Energiecentrales staat in het blauw het maximum dat ze kunnen leveren.

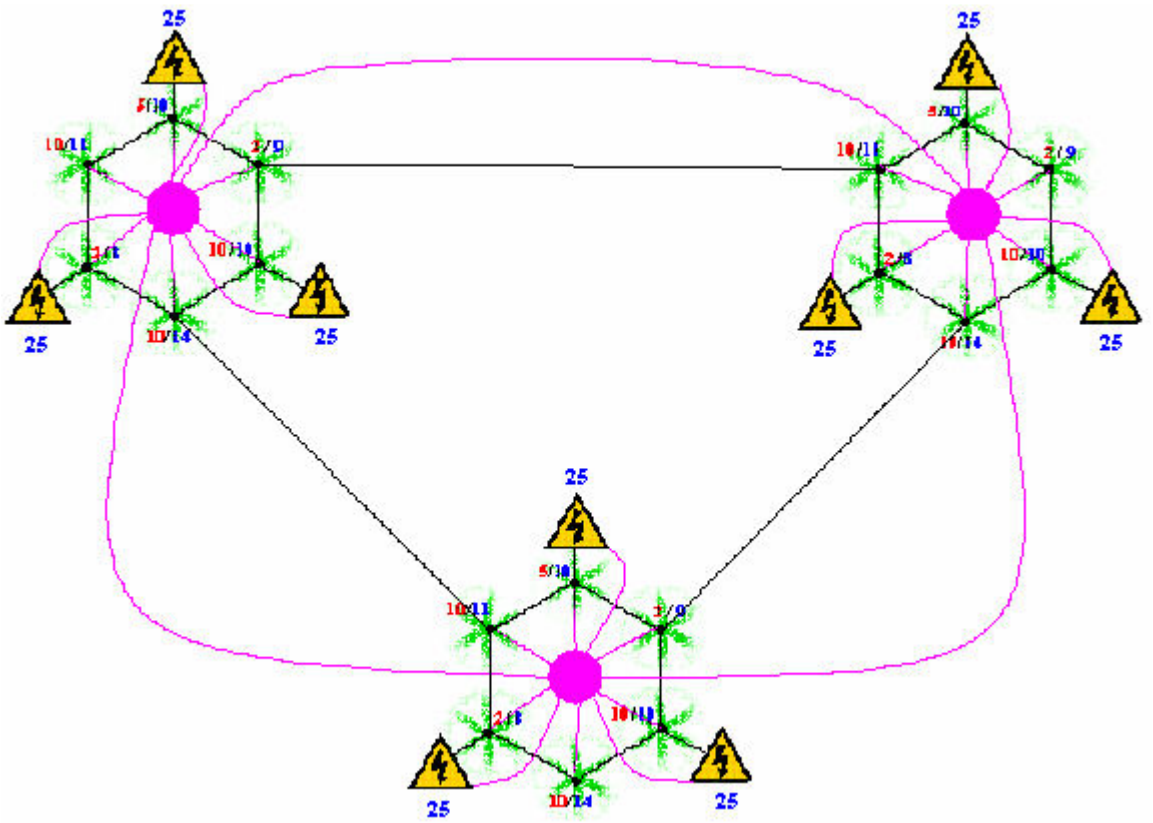
In het normale geval leveren de Energiecentrales net zoveel als de totale vraag. In figuur 1a is te zien wat iedere Gebruiker vraagt en met welke prioriteit. Deze Vraag en Prioriteit worden gebundeld in de *Nodes*, die onderling tot een verdeling komen van de stroom. In figuur 1b is te zien wat de Vraag en Prioriteit van de *Nodes* zijn geworden. De Vraag is opgeteld wat de Gebruikers willen (kunnen ook andere *Nodes* zijn!) en de Prioriteit is het maximum van de prioriteiten van alle Gebruikers die (in)direct voorzien worden van stroom door de *Node*.



figuur 1b: de Prioriteiten en Vraag van de Nodes

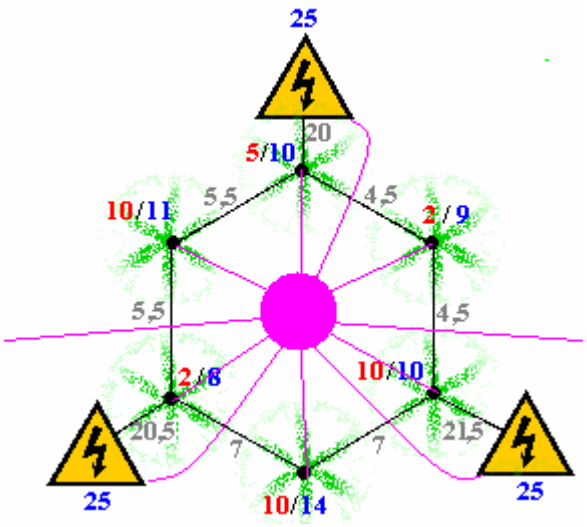
Hogere prioriteit gaat natuurlijk voor. De *Node* linksonder (die van **2**/**8**) zal de stroom eerst doorleiden naar de *Nodes* van **10**/**11** en **10**/**14**. Zodra deze *Nodes* genoeg stroom hebben gekregen voor hun Gebruikers met Prioriteit 10 zal de Prioriteit van de *Nodes* veranderen, de Gebruiker met Prioriteit 10 is voorzien. De Prioriteit van de *Node* zal dus lager worden, en er zal dan bekeken worden welke *Nodes* (en hun Gebruikers) daarna stroom krijgen. Als een *Node* een Gebruiker heeft met Prioriteit 10 en een andere *Node* vraagt om stroom met Prioriteit 10, zal eerst de eigen Gebruiker stroom gegeven worden en daarna, als er stroom over is, stroom doorgegeven worden aan de andere *Node*. Zo wordt op basis van Prioriteit het hele netwerk van stroom voorzien.

Het regelen van de Verbindingen waar een zekere hoeveelheid stroom overheen loopt, is een goede taak voor de hogere intelligente laag. Deze laag moet de optimale verdeling bepalen. Optimaal betekent hier een zo evenredig mogelijke belasting van de Verbindingen in het netwerk en dat zo veel mogelijk aan de Vraag van de Gebruikers voldaan wordt. De optimale verdeling wordt doorgegeven aan de *Nodes*. De *Nodes* moeten deze verdeling aanhouden. Als er nu een verbinding kapot gaat, kan dat lokaal snel opgevangen worden, en ondertussen gaat de hogere laag op zoek naar een nieuwe verdeling. Zodra deze gevonden is, zal dat doorgegeven worden aan de *Nodes* die zich dan weer aanpassen.



figuur 1c: het hogere intelligente netwerk

Uiteraard is dit slechts mogelijk wanneer elke *Node* een aantal verbindingen heeft met andere *Nodes* en misschien ook meerdere groepen Gebruikers. Een representatie van deze hogere laag is te zien in figuur 1c. Hierin stellen de roze cirkels communicatiecentra voor, en de roze lijnen communicatielijnen. Hierover kunnen informatie en commando's verstuurd worden. In figuur 1c hebben de Verbindingen in een lokaal netwerk een maximum capaciteit van 15 stroomeenheden. De draden van de Energiecentrale naar een *Node* hebben een maximum capaciteit van 25, en de Verbindingen tussen de clusters hebben ook een verhoogde capaciteit, om het voorbeeld eenvoudig te houden. Door de maximum capaciteit van de Verbindingen is het goed om een optimale verdeling van de stroom te hebben in het netwerk. Door de stroom optimaal te verdelen over de leidingen kan de resterende capaciteit direct gebruikt worden in geval van problemen. Ook wordt de gemiddelde levensduur van de stroomkabels verlengd. Voor 1 lokaal netwerk is de load uitgewerkt in figuur 1d. De load van de Verbindingen is in het grijs te zien. De *Nodes* kunnen hier gewoon gebruiken wat hun Vraag is.



figuur 1d: de Load v/d Verbindingen

Wiskundig model

Om te bepalen hoe effectief de oplossingen zijn die een methode geeft is de volgende formule te gebruiken op deze oplossingen:

$$score = \sum_{i=0}^{n} p_i \cdot \frac{s_i}{s_{vraag}}$$

Hierbij geeft i de categorie(in onze case huizen/industrie/ziekenhuis) van de gebruiker aan waarbij p_i staat voor de prioriteit van de categorie en s_i voor de hoeveelheid stroom die een categorie gebruikt. s_{vraag} staat dan voor de totale vraag om stroom in het netwerk. Bij deze formule behaal je een hoge score door de gebruikers met een hoge prioriteit de meeste stroom te geven en door te zorgen dat er geen stroomtekorten zijn.

Als voorbeeld kunnen we de score uitrekenen van het netwerk in figuur 1b:

- De totale vraag van het netwerk: $s_{vraag} = 35+17+10 = 62$
- Het stroomgebruik van de huizen categorie (prioriteit 2): $s_1 = 17$
- Het stroomgebruik van de industrie categorie (5): $s_2 = 10$

- Het stroomgebruik van de hospitaal categorie (10): $s_3 = 35$

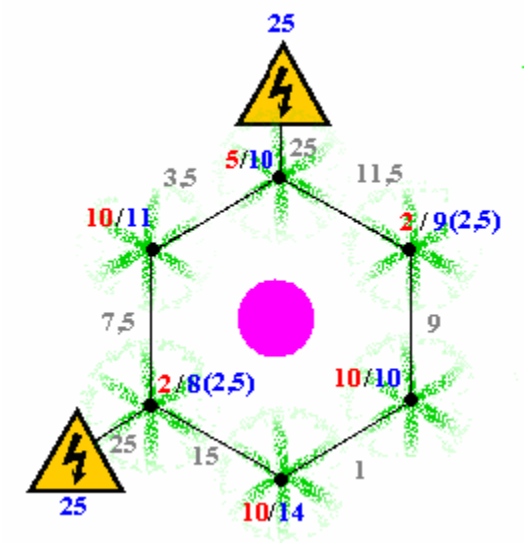
Dus: score = 2*17/62 + 5*10/62 + 10*35/62 = 7

Omdat alles stroom krijgt, is 7 voor dit netwerk de maximale score. Als er een elektriciteitscentrale uitvalt, kunnen niet alle gebruikers zoveel stroom gebruiken als ze vragen, omdat er niet genoeg is. Dan zal de score lager uitvallen, en de beslissingsmethode moet proberen de maximale score te behalen.

2.2 Scenario's

Scenario 1: Energiecentrale valt uit

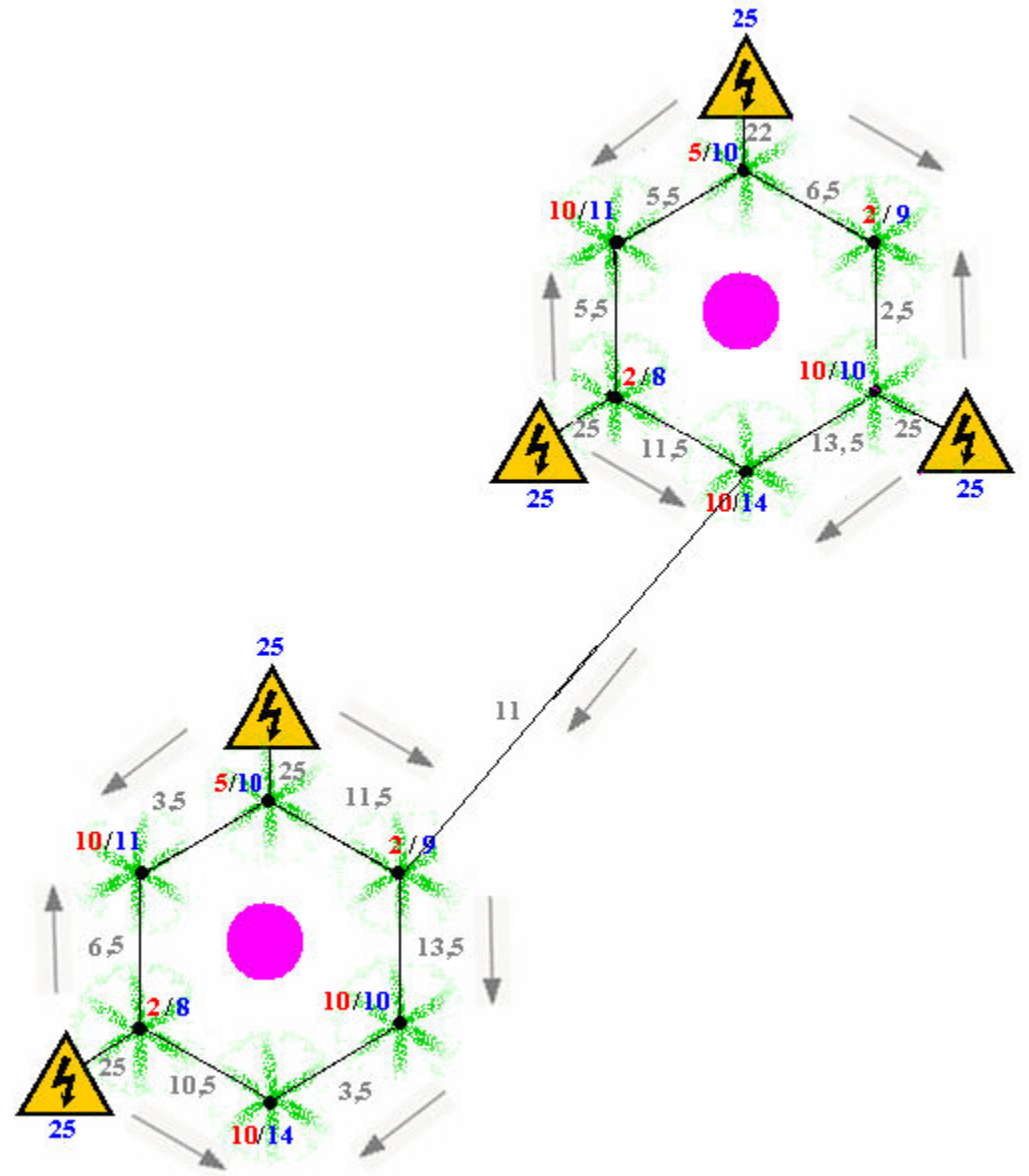
Als er een centrale uitvalt, onstaat er een stroomtekort. De overgebleven Energiecentrales leveren nu niet genoeg stroom voor de lokale cluster, dus er moeten maatregelen genomen worden. In figuur 2a is een cluster met een uitgevallen Energiecentrale gevisualiseerd.



figuur 2a: een Energiecentrale is uitgevallen; de stroomverdeling is lokaal aangepast. Voor de duidelijkheid zijn de communicatiekanalen niet weergegeven.

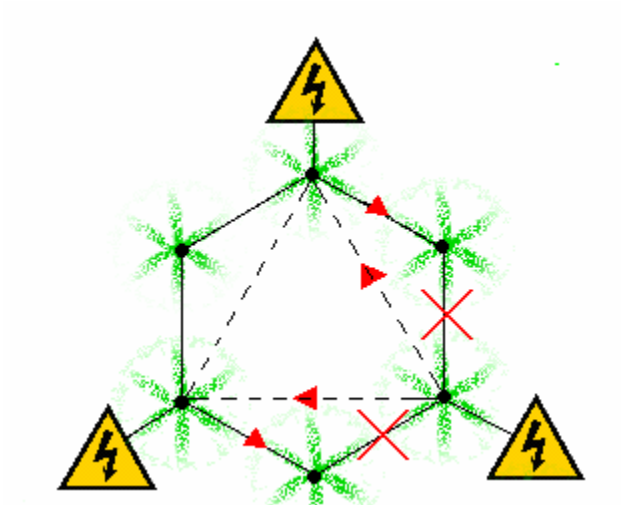
In figuur 2a zijn er twee *Nodes* te zien die een tekort hebben: Gevraagd is **8**, gekregen is **2,5** en gevraagd is **9**, gekregen is **2,5**. Deze *Nodes* hebben niet voldoende stroom gekregen, omdat zij een lage Prioriteit hebben. Het is nu de taak van het hogere intelligente netwerk om stroom uit andere clusters te halen, zodat het tekort opgeheven wordt. In dit geval nemen we even aan dat de probleemcluster de onderste is in figuur 1c, en dat hulp wordt gevraagd aan de cluster rechtsboven. De cluster rechtsboven krijgt van het hogere netwerk bijvoorbeeld de opdracht om de twee onderste centrales 25 eenheden te laten produceren en de bovenste 22, en de extra stroom naar de probleemcluster te transporteren. De twee clusters zullen er dan vervolgens uit gaan zien als figuur 2b.

De gevonden verdeling hoeft niet optimaal te zijn qua belasting van het netwerk. Dit is iets waar meer tijd aan besteed moet worden. In eerste instantie wordt dus het tekort opgeheven. Als het hogere netwerk na verloop van tijd een betere oplossing heeft gevonden voor de stroomverdeling, zal dit doorgegeven worden aan het netwerk.



figuur 2b: extra stroom; de richting waarin de stroom stroomt, is aangegeven met de pijlen.

Scenario 2: Gebroken elektriciteitsdraden
Een ander scenario is dat er een of meerdere Verbindingen breken. Dit tast de structuur van het netwerk aan. Er moet een nieuwe route gevonden worden om de stroom door het netwerk te transporteren. Hiervoor kunnen ongebruikte Verbindingen (stippellijnen) aangeschakeld worden. In figuur 2c is de situatie weergegeven. Een gedeelte van de stroom zal dan over de nieuwe Verbinding gaan lopen om de onderste *Node* van stroom te voorzien. Een ander gedeelte van de stroom kan via de andere nieuwe Verbinding stroom rechtsboven leveren. Het is voor een dusdanige oplossing wel noodzakelijk dat er Verbindingen liggen die niet of nauwelijks gebruikt worden. Een omleiding voor de stroom kan ook een oplossing zijn als een *Node* teveel stroom moet doorgeven. Dan kan er en andere route gevonden worden die op dezelfde plaats aankomt, maar een ander pad gebruikt.



figuur 2c: Twee Verbindingen zijn gebroken

Scenario 3: Communicatiestoornis
Een laatste scenario wat hier besproken zal worden is het uitvallen van communicatie tussen clusters en de hogere netwerklaag die hen eventueel aanstuurt. Clusters moeten nu zelf al hun problemen oplossen.

3. Beslissingsmethoden

Problemen die zich voordoen in het netwerk dienen opgelost te worden. Een oplossing kan met verschillende methodes tot stand komen. Ook zijn er methoden expliciet geschikt voor analyse van het netwerk, ofwel te achterhalen wat de oorzaak is van het probleem. Dit is iets anders dan het uitzoeken hoe de stroom moet gaan lopen als reactie op het probleem, maar kan eventueel wel als basis dienen voor het (tijdelijk) oplossen van het probleem.

Uiteraard zijn er nog veel meer methodes en varianten beschikbaar die interssant zouden kunnen zijn. Het is onmogelijk alle methodes en varianten te bespreken. Hieronder worden een aantal interessante en veel voorkomende beslissingmethoden besproken.

3.1 Rule-based

De Rule-based methode is gebaseerd op eenvoudige regels die als volgt zijn opgebouwd:
regelnaam: **als** <conditie> **dan** <actie>
Hierbij is de regelnaam de naam van de regel en <conditie> is de voorwaarde waarop een bepaalde <actie> wordt uitgevoerd. Zowel <conditie> als <actie> kunnen bestaan uit meerdere onderdelen.

Om het geheel wat overzichtelijker te maken zijn de drie communicatienodes als een afzonderlijk systeem genomen. Voor alle communicatienodes is de volgende regel van toepassing:

```
all: als som(Vraag_1) > som(Capaciteit_1)
      EN som(Vraag_2) < som(Capaciteit_2)
      EN som(Vraag_3) < som(Capaciteit_3)
            dan sturen 2 en 3 extra stroom.
```

Zo ook voor een tekort bij knooppunten 2 of 3 en bij meerdere knooppunten met een tekort en dat geeft al 6 regels. Deze regels houden nog geen rekening met de verschillende prioriteiten waardoor dit dus nog niet eens optimaal systeem aflevert. Om het systeem optimaal te maken is er een zeer groot aantal regels nodig: namelijk voor elk mogelijk probleem.

Op lager en globaal niveau is het nog moeilijker om regels te maken die direct actie ondernemen. Het is mogelijk om alle verbindingen om een *Node* heen 'a', 'b' of 'c' (verbinding naar een centrale) te noemen en vervolgens regels als:

```
verdeelpunt1: als a == defect dan gebruik b
verdeelpunt2: als b == defect dan gebruik a
verdeelpunt3: als capaciteit(a) < vraag EN capaciteit(b) >
capaciteit(a) dan gebruik b
verdeelpunt4: als som(capaciteit(a),capaciteit(b)) > vraag
      EN capaciteit(a) < vraag EN capaciteit(b)
            dan gebruik a EN gebruik b

centrale: als vraag > capaciteit dan stuur bericht naar hoofdnode
```

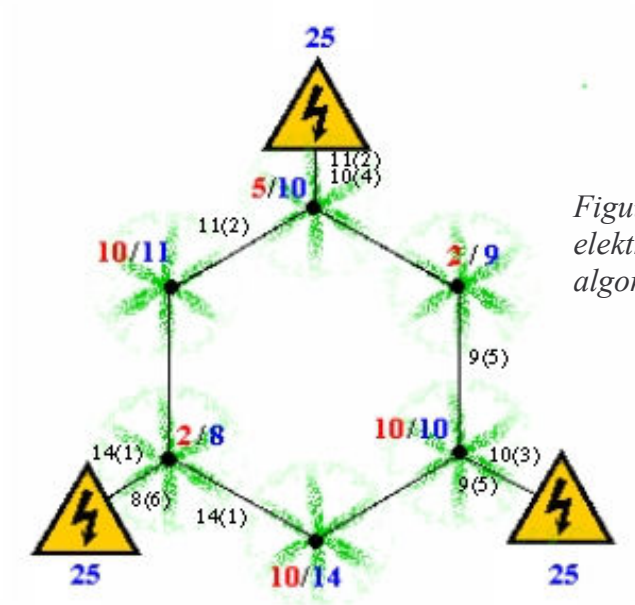
Het is lastig om dit systeem in regels te vatten omdat het systeem te veel van elkaar afhankelijk is en er eigenlijk teveel acties genomen moeten worden als er iets mis gaat. Dit is deels op te lossen door het probleem door te sturen naar een andere *Node* in het netwerk of naar een *Node* in het hogere netwerk.

De schaalbaarheid van een Rule-based systeem is een probleem, want voor een groot systeem zijn er veel te veel regels nodig. Als er een onderdeel toegevoegd wordt, moet er gelijk ook een grote hoeveelheid extra regels aan het systeem toegevoegd worden.

3.2 Gradient

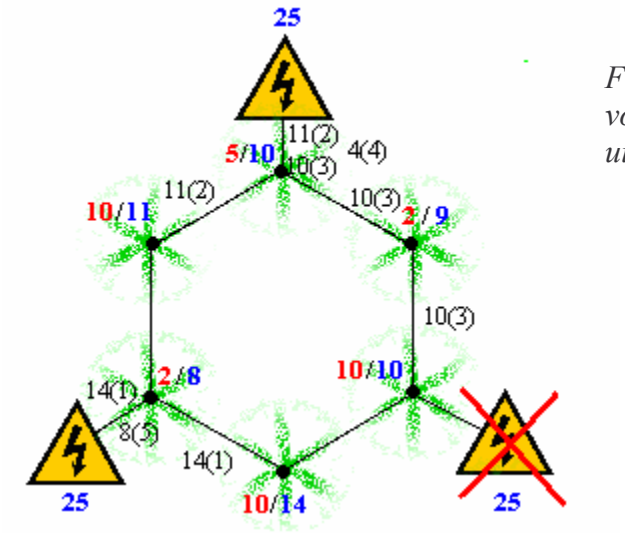
Het doel van de gradient methode is het zoeken van de kortste route van een *Node* naar een elektriciteitscentrale die nog elektriciteit kan leveren. Een extra probleem is de maximale capaciteit die de stroomleidingen kunnen verwerken. Hoe zwaarder een leiding belast wordt, hoe hoger de kosten om er meer elektriciteit doorheen te sturen. Het systeem zal gaan zoeken naar een optimum, maar dat zal enige tijd duren omdat voor iedere *Node* steeds een nieuwe optimale route moet worden gezocht, aangezien de routes invloed hebben op elkaar.
De kosten om stroom te transporteren door een leiding worden dus bepaald door de hoeveelheid stroom die er al doorheen gaat. Daarnaast telt iedere *Node* die gepasseerd

moet worden om het doel te bereiken als extra kosten. De prioriteit bepaalt welke *Node* als eerste een gradient berekening mag uitvoeren. Een *Node* met een hoge prioriteit zal daardoor vrijwel zeker stroom krijgen. Als de prioriteit hetzelfde is dan gaat de *Node* met de grootste vraag voor. In de beginsituatie zal figuur 3a ontstaan.



Figuur 3a: de verdeling van de elektriciteit volgens het gradient algoritme.

De cijfers tussen de haakjes geeft de volgorde van berekenen aan: (1) is dus als eerste berekend. Het cijfer voor de haakjes geeft aan hoeveel elektriciteit er wordt getransporteerd over de leiding. Het *Node* met waarde 2/9 kan van 3 plaatsen elektriciteit krijgen, maar de centrale linksonder valt eigenlijk al meteen af omdat deze te ver weg is. Dan blijven er nog 2 over die interessant zijn om uit te rekenen.
De centrale boven heeft al een stroom lopen van 11 + 10 = 21 naar het eerste *Node*, dus de kosten om daarover nog meer stroom te laten lopen is veel hoger dan bij de centrale rechtsonder, die maar 10 stroom heeft lopen. De afstand tussen de *Node* en de centrale is in beide gevallen hetzelfde, dus zal er gekozen worden voor de centrale rechtsonder. Als voor de kosten de waarde wordt genomen die al over de leiding heen gaat + 1 per *Node*, dan zou de route naar de bovenste centrale 11+ 10 + 1 = 22 zijn en de centrale rechtsonder 10 + 1 = 11.



Figuur 3b: de verdeling van de elektriciteit volgens het gradient algoritme na het uitvallen van de centrale rechts-onder.

Het probleem is dat de gradient methode maar over één leiding de stroom kan aanvoeren en niet over meerdere leidingen. Hierdoor is het vrij moeilijk om alle systemen van stroom te voorzien als er iets mis gaat.

3.3 Bayesiaans netwerk

Met een Bayesiaans netwerk kunnen problemen die binnen het sensornetwerk optreden geanalyseerd worden. Als de oorzaak van een probleem bekend is, kan daar makkelijker een oplossing voor gevonden worden. Het Bayesiaanse netwerk representeert het domein. Iedere knoop is een random variabele met een bijbehorende voorwaardelijke kanstabel, waarin de effecten van de 'ouders' worden verwerkt. Als een knoop geen 'ouders' heeft, is er een apriori kans op het optreden van dat verschijnsel. Een pijl geeft een directe invloed aan van de ene random variabele op de andere. Bij het optreden van een bepaald verschijnsel, kunnen de voorwaardelijke kansen van dat verschijnsel met mogelijke oorzaken berekend worden. De oorzaak met de hoogste kans wordt dan aangenomen als oorzaak van het verschijnsel. Zodra de oorzaak bekend is, moet er nog een oplossing worden gezocht. Dit kan niet met behulp van het Bayesiaanse netwerk. Een ander nadeel is dat de kansen van het voorkomen van bepaalde verschijnselen meestal niet precies bekend zullen zijn en dus geschat moeten worden.

3.4 Auctions

Veilingen zijn een manier om goederen te verdelen over verschillende gegadigden op een zo efficiënt mogelijke manier. Door de eeuwen heen zijn er verschillende soorten veilingen uitgedacht waardoor er voor de meeste situaties wel een veiling is. Zo is er het Engelse systeem: een veilingmeester veilt de goederen en de kopers mogen hierop zo vaak bieden als ze willen. Degene met het hoogste bod krijgt het goed tegen de geboden prijs. Ook is er het Nederlandse systeem waar de waarde afneemt tot iemand biedt en zo de veiling wint. Een andere bekende veilingmethode is het Vickrey systeem waar de bieders maar één keer mogen bieden en dit blind doen (ze weten niet wat de anderen bieden). De winnaar moet de prijs betalen van het op een na hoogste bod.
In het elektriciteitsnetwerk dat we bestuderen moet de stroom worden verdeeld over de verschillende gebruikers, waarbij geldt dat bepaalde gebruikers een hogere prioriteit hebben dan anderen om de aanwezige stroom te krijgen. Deze prioriteit kunnen we goed gebruiken bij het veilingsysteem door ermee te beslissen welk bod wint en welke gebruiker de stroom dus krijgt.
Bij een systeem waarvoor geldt dat de prioriteit omhoog gaat wannneer een systeem niet wint bij het bieden, kan er ook gewerkt worden met een cumulatieve hoeveelheid geld: wie lang niet wint geeft niks uit en zal dus ooit ook aan de beurt komen. Bij het elektriciteitsnetwerk hebben we dit in het huidige systeem echter niet nodig doordat

gebruikers met een hoge prioriteit altijd belangrijker zijn dan gebruikers met een lage prioriteit.
Van de verschillende soorten veilingen die er bestaan lijkt de beste keus voor het elektriciteitsnetwerk een versie van het Vickrey systeem waar dus alle systemen maar één keer bieden en de hoogste daaruit wordt gekozen. Hoeveel de winnaar moet betalen is niet van belang omdat, zoals eerder gezegd, de hoeveelheid geld niet cumulatief is.

3.4.1 Centrale veiling

Als het elektriciteitsnetwerk helemaal goed werkt en de communicatienodes goed werken en alle informatie van het netwerk hebben (*full observability*) kan er een van deze *Nodes* als veilingmeester fungeren en met behulp van het veilingsysteem een optimale verdeling van de stroom maken. Dit systeem is ook behoorlijk flexibel en kan het goed omgaan met nieuwe of verdwijnende gebruikers of leveranciers.

Zo'n centrale veiling zou op de volgende manier gaan:

1. Op basis van de gegevens van alle stroomleveranciers wordt vastgesteld hoeveel stroom er te verdelen is (het aanbod).
2. Alle gebruikers van de stroom geven hun vraag en prioriteit door (het bod).
3. De veilingmeester stelt vast welke gebruikers het hoogste bod hebben ingediend (in ons voorbeeld zullen dat er meerdere zijn) en wijst de gevraagde stroom toe. Als blijkt dat er niet genoeg stroom is voor deze winnaars wordt de stroom verdeeld volgens bepaalde regels (bijvoorbeeld ieder een gelijk percentage of als ze alles of niets nodig hebben een willekeurige verdeling).
4. De toegewezen stroom wordt van het aanbod afgetrokken. Als er nog iets over is begint de veiling opnieuw.

3.4.2 Decentrale veiling

Als nu de centrale veiling uitvalt en de gebruikers zelf de stroom moeten gaan verdelen hoeft dit niet meteen tot een optimaal resultaat te leiden. Het systeem na een bepaalde tijd wel een zo goed mogelijke verdeling voortbrengen. Een algoritme om dit te bereiken zou het volgende kunnen zijn:

1. De leveranciers van de stroom veilen de stroom aan hun directe buren volgens het systeem dat ook bij globale veilingen wordt gebruikt. Zij onthouden daarbij echter of er, en zo ja, bij welke gebruikers een tekort ontstaat.
2. De gebruikers die door de nieuwe verdeling stroom krijgen veilen dit op hun beurt weer aan hun buren (waaronder ook de leverancier van hun stroom valt). Hierbij geldt dat alle inkomende stroom geveild wordt en dat ze dus ook zelf meedoen aan hun eigen veiling.
3. Als een gebruiker in zijn eigen veiling heeft geconstateerd dat een van de deelnemers een tekort had zal hij proberen bij de volgende veiling waaraan hij deelneemt wat extra stroom te kopen zodat ook systemen met een tekort stroom krijgen. Dit doet hij door een extra bod uit te brengen voor de gebruiker met een tekort (behalve als deze gebruiker zelf meedoet aan de veiling). Om te voorkomen dat er door de buren van de Gebruiker met het tekort teveel stroom bij elkaar

wordt geregeld zal dit bod maar een deel van het tekort zijn. Ook is het misschien beter om systemen die direct aan de veiling meedoen een voordeel te geven door bij de indirect biedene systemen de prijs (prioriteit) één stapje lager in te stellen dan het originele bod zodat een rechtstreeks bod met dezelfde prioriteit voorgaat.

De veilingen zouden zich met dit systeem door het netwerk verspreiden en om te zorgen dat het aantal veilingen niet blijft toenemen moeten er misschien regels zijn waarbij bijvoorbeeld geldt dat een gebruiker maar in één veiling tegelijk mag bieden, hij maar één keer in de zoveel tijd een veiling mag starten ed.
Op deze manier worden de tekorten langzaam maar zeker door het netwerk gepropageerd en zal uiteindelijk een equilibrium ontstaan dat het dichtst bij de optimale verdeling ligt. De grootte van de afstand tussen een overschot en een tekort (aantal gebruikers er tussenin) is evenredig met de tijd die het kost om het overschot goed te verdelen.

De centrale en decentrale veilingen kunnen op verschillende lagen worden toegepast. Zo kun je bij het voorbeeld met drie clusters van gebruikers die via hun centrale (roze) *Nodes* met elkaar verbonden zijn, deze supernodes (roze) met de centrale veilingen de stroom over de cluster laten verdelen waarbij de drie supernodes onderling de stroom weer verdelen met het decentrale systeem. Als dan een supernode uitvalt kunnen de gebruikers het decentrale systeem gebruiken om toch nog onderling de stroom zo optimaal mogelijk te verdelen.

3.5 *Distributed Breakout*

Het *Distributed Breakout Algorithm* is een *greedy best-first* zoekalgoritme, met een mogelijkheid om uit een lokaal minimum te ontsnappen. Het algoritme gebruikt als maat van correctheid het aantal fouten dat er nog in de huidige configuratie zit. Volgens de *greedy best-first* methode kijkt het algoritme of er een aanpassing te maken is op de huidige configuratie die resulteert in een verminderd aantal fouten, zodat de foutscore lager is. Wanneer het algoritme in een lokaal minimum zit, zullen de fouten die op dat moment nog in de configuratie zitten voortaan zwaarder gaan meetellen, zodat de foutscore hoger wordt. Op die manier wordt het lokale minimum “opgeheven” en kan het algoritme verder zoeken naar de optimale configuratie. Elk lokaal minimum dat het algoritme tegenkomt op weg naar het globale minimum wordt zo dus opgeheven, en wanneer het algoritme in een globaal minimum zit, kunnen de fouten in principe oneindig opgehoogd worden. Het algoritme krijgt er alleen maar fouten bij als het algoritme van dit globale minimum afwijkt, dus het zal in die configuratie blijven.
In het elektriciteitsnetwerk is een configuratie de manier waarop de stroom verdeeld is en het beginpunt de configuratie die door een lokaal netwerk is bepaald. Deze beginconfiguratie is vaak niet de optimale configuratie, daar is het hogere intelligente netwerk voor om die op te zoeken door middel van dit algoritme. Elke kleine aanpassing/rerouting van hoe de stroom loopt kan een positief/negatief effect hebben, dus een verandering in score opleveren. Pas als de optimale configuratie gevonden is, zal dit doorgegeven worden aan het lokale netwerk, zodat die de configuratie aan kan nemen. In het zoekproces worden namelijk meerdere suboptimale configuraties getest, en die kunnen natuurlijk niet allemaal door het netwerk geprobeerd worden.

Dit algoritme kan relatief veel tijd & rekenkracht kosten, dus daarom is het allen geschikt voor het hogere netwerk. Het algoritme is geschikt voor het zoeken van een nieuwe configuratie, maar helaas niet zo snel. Ook is het algoritme niet geschikt voor de analyse van een netwerk; het zal geen problemen op kunnen sporen. het *Distributed Breakout* algoritme is echter wel geschikt voor een groot, dynamish netwerk. Het zoekt een optimale configuratie door een *greedy best-first* zoekactie uit te voeren in de ruimte van alle mogelijke configuraties. Het toevoegen van één of meerdere Gebruikers zal wat rekentijd betreft relatief weinig uitmaken.

3.6 Neurale Netwerken

Een neuraal netwerk zou, mits goed getraind, heel goed als beslissingsmethode kunnen werken. Neurale netwerken vallen echter direct af omdat er geen mogelijkheid tot trainen is. Het netwerk zou getraind kunnen worden in een simulatie. Het is alleen onmogelijk alles te simuleren wat mis zou kunnen gaan in het netwerk. Als er iets onverwachts misgaat, kan het neurale netwerk ook onverwachte of zelfs ongschikte beslissingen nemen. In het geval van een elektriciteitsnetwerk kan men zich dergelijke fouten niet veroorloven.

3.7 Fuzzy Logic

Fuzzy logic lijkt sterk op rule-based methoden met als extra een geloofswaarde aan iedere regel. Het enige voordeel hiervan is dat fuzzy logic gebruikt kan worden om een analyse te maken van het probleem. In het geval van het elektriciteitsnetwerk zijn er echter geen fuzzy begrippen, er word alleen maar gebruikt gemaak van concrete waarden. In de hoger liggende lagen heeft deze methode teveel regels nodig net als bij rule-based om te functioneren.

4. Eigenschappen

In deze paragraaf worden verschillende eigenschappen van een aantal beslissings-methoden besproken. Ook worden scenario's genoemd waarvoor deze eigenschappen van belang zijn. Tot slot wordt er op basis van de genoemde eigenschappen een voorstel gedaan voor methodes om de mogelijke problemen in het elektriciteitsnetwerk zo goed mogelijk op te lossen.

Gedistribueerd
Is de methode gedistribueerd? Een gedistribueerde methode heeft geen centraal punt waar alles uitgerekend wordt. Als de methode niet gedistribueerd is en het centrale punt valt uit, kunnen er geen beslingen meer worden genomen in het netwerk. In het ongunstigste geval ligt dan het hele netwerk plat. Gedistribueerde methodes zijn niet afhankelijk van een centraal punt. Als er delen uitvallen wordt er toch de 'beste' beslissing gemaakt op basis van wat wel bekend is. Men spreekt in deze context ook wel van *graceful degradation*: als een gedeelte uitvalt, ligt niet meteen het hele systeem plat. Het nog wel

werkende deel van het systeem opereert zo goed mogelijk en probeert het probleem op te vangen.
Gedistribueerde systemen zijn van belang voor alle scenario's.

Analyse
Maakt de methode een analyse van het probleem? Met andere woorden: wordt er geprobeerd de precieze oorzaak van een fout te achtehalen? Dit kan helemaal losstaan van de eventueel te ondernemen actie, maar het kan ook dienen als extra informatiebron om een zo goed mogelijke actie te kiezen.
Analyse komt niet terug in de scenario's, wel is het handig dat bepaald kan worden wat er kapot is zodat en reperateur gestuurd kan worden om het te repareren. Deze eigenschap is opgenomen voor de compleetheid.

Actie
Zoekt de methode een actie om het probleem op te lossen?
In Scenario 1 en 2 moet er sowieso een actie ondernomen worden, in scenario 3 is dit niet altijd nodig er is alleen geen communicatie meer. Mocht er naast de communicatie nog wel iets misgaan dan moet wel actie ondernomen worden.

Globaal en Lokaal
Het systeem heeft verschillende niveaus. Het globale niveau omvat het hele netwerk. Globaal betekent dus of de methode geschikt is om het hele netwerk aan te sturen (de hogere intelligentie). Het Netwerk is opgebouwd uit clusters, die op hun beurt weer opgebouwd zijn uit *Nodes*. Het lokale niveau kan gedefinieerd worden als de clusters of als de afzonderlijke *Nodes*. Omdat deze lokale niveaus verschillende eigenschappen hebben, hebben wij ze uitgesplitst in het Nodeniveau en het Clusterniveau.
Deze eigenschappen zijn niet relevant voor de scenario's.

Snel
Snel geeft aan of een methode op een relatief snelle manier tot een oplossing komt.
Deze eigenschap is niet relevant voor het vinden van een oplossing.

Schaalbaar
Bij een schaalbare methode gaat de performance niet (teveel) achteruit als het systeem (veel) groter wordt.

Routing
Bestaat de oplossing van het probleem uit het vinden van een nieuwe route voor (in dit geval) de stroom?
Routing heeft voornamelijk te maken met scenario 2, maar ook in scenario 1 zal misschien een andere route voor de stroom gevonden moeten worden.

Optimale verdeling
Komt de methode uiteindelijk tot een globale optimale verdeling? We nemen dus aan dat er een manier is om te berekenen of een bepaalde verdeling optimaal is. Deze optimale

verdeling hoeft niet de enige oplossing te zijn; er kunnen nog een hoop sub-optimale oplossingen zijn. Taak van het systeem dus om een optimale oplossing te vinden.
Met scenario 1 en 2 kan getest worden of er een optimale verdeling gevonden wordt.

Communicatie Onafhankelijk
Is de methode onafhankelijk van communicatie tussen *Nodes* onderling en tussen *Nodes* en de hogere intelligente laag? Als dit zo is, kunnen de *Nodes* nog enigszins intelligente beslissingen nemen als het hogere netwerk uitvalt.
Scenario 3 bepaalt vooral of de methode communicatie onafhankelijk is.

	Gedistribueerd	Analyse	Actie	Lokaal (nodeniveau)	Lokaal (clusterniveau)	Globaal	Snel	Schaalbaar[3]	Routing	Optimale verdeling	Communicatie Onafhankelijk (scenario 3)
Rule-based	+	-	+	+	+	-	+	-	-	-	+
Gradient	+/-[1]	-	+	-	+	+	+	-	+	+	+
Bayes	-	+	-	-	+	-	+	-	-	nvt	+
Centrale Veiling	-	-	+	-	+	+	+	+	+	+	-
Decentrale Veiling	+	-	+	+	+	+	-	+	+	+	-
Distributed Breakout	+	-	+	+	+	+	-	+	+	+	-
Neuraal net	-	-	+	+	?	?	+/- [2]	-	-	-	-
Fuzzy Logic	+	+	+	+	+	-	+	-	-	-	+

Tabel 1: De methodes en hun mogelijk gebruik voor self-reconfigurable systemen (+ = geschikt; - = niet geschikt; ? = onbekend; nvt = niet van toepassing).

1: Om een Gradient te berekenen is er een kaart nodig van het hele netwerk
2: Neurale netwerken zijn snel in het geven van een uitkomst, maar de training duurt erg lang.
3: Het aantal regels van een Rule-based en Fuzzy logic neemt sterk toe bij een kleine toevoeging aan het netwerk. Gradient moet van het hele systeem een 'kaart' hebben, om de beste route op te zoeken en Bayes moet voor elk onderdeel in het netwerk een apart object hebben, met een zekere kans dat het object werkt/kapot is. Het bayesian netwerk wordt dus erg groot als het netwerk ook groot is. Het neurale netwerk moet zeer lang leren voor een groot netwerk en als het netwerk wordt aangepast moet het alles helemaal opnieuw leren.

Voorgestelde oplossing
Uit de genoemde eigenschappen blijkt dat geen enkele methode snel een optimale oplossing kan geven. Een combinatie van een snelle en een optimale methode lijkt een goed alternatief. De Gradient methode geeft snel een suboptimale oplossing die gebruikt kan worden, zolang een langzamere methode zoals een veiling of *Distributed Breakout* nog geen oplossing gevonden heeft.

Implementatie van het netwerk en verschillende beslissingsmethodes zou kunnen helpen bij het bepalen van een goede combinatie. Het netwerk en de problemen zijn te complex om met de hand door te rekenen. De reacties van het systeem op problemen zijn moeilijk te voorspellen. Ook zijn veel zaken die direct of indirect invloed hebben op de prestaties van de verschillende beslissingsmethodes, zoals grootte van het netwerk, clustergrootte, communicatiesnelheid, beschikbare rekenkracht (in bijvoorbeeld de sensoren) en voor deze case specifiek het aantal Energiecentrales en verbindingen en de capaciteit van deze vebindingen van invloed.

5. Conclusie

In dit artikel is een aantal beslissingsmethoden besproken voor een gedistribueerd *self-reconfigurable* elekrticiteitsnetwerk. Eigenschappen van deze methodes zijn samengevat in tabel 1. Het is belangrijk om te onthouden dat in het elektriciteitsnetwerk de verbindingen waar stroom overheen loopt niet gelijk zijn aan communicatielijnen.

Helaas is er niet één methode geschikt om zowel op globaal als lokaal niveau alle problemen optimaal op te lossen. In tabel 2 is samengevat welke van een aantal veelbelovende beslissingsmethodes geschikt zijn om de verschillende scenario's op te lossen. In de tabel wordt ook de Bayesiaanse methode genoemd ook al is dit geen beslissingsmethode. Een Bayesiaans netwerk is wel uitermate geschikt om een analyse te geven van een probleem. Die analyse kan gebuikt worden als invoer voor een beslissingsmethode.

	Scenario 1: Het uitvallen van een energiecentrale	Scenario 2: Gebroken elektriciteitsdraden	Scenario 3: Communicatistoornis
Rule-based	+	+	+
Gradient	+	+	-
Bayes	+	+	+
Centrale Veiling	+	+	-
Decentrale Veiling	+	+	-
Distributed Breakout	+	+	-

Tabel 2: Geschiktheid van een aantal methoden voor het oplossen van de scenario's

In tabel 2 wordt uitgegaan van het netwerk in figuur 1c. Eventuele problemen met schaalbaarheid worden dus niet meegenomen.

Rule based is in principe te gebruiken met alledrie de scenario's. Bij grotere netwerken ontstaan er echter problemen met de schaalbaarheid. Het aantal benodigde regels zal veel te goot worden.

In scenario's 1 en 2 moet er een nieuwe verdeling van de stroom over het netwerk worden gevonden. De Gradient methode vindt snel een oplossing, maar die is niet optimaal. Scenario 3 is niet op te lossen, voor de Gradient methode is namelijk communicatie nodig.

Veilingen zijn ook afhankelijk van communicatie. Voor het verdelen van stroom is het een geschikte methode. Omdat het hier om een gedistribueerde systemen gaat, lijkt een gedistribueerde veiling de beste oplossing. Het duurt dan echter wel lang voor een optimale oplossing is gevonden.

De *Distributed Breakout* methode is ook afhankelijk van communicatie. Ook duurt het even voor een optimale oplossing gevonden is.

Communicatie is een zwak punt van beslissingsmethodes voor gedistribueerde *self-reconfigurable* systemen.

Alle methodes hebben dus sterke en zwakke kanten. Het ligt daarom voor de hand een mix van methodes te gebruiken voor het elektriciteitsnetwerk. Implementatie van het netwerk en verschillende beslissingsmethoden kunnen helpen bij het maken van goede keuzes. Op basis van de theorie is te concluderen dat het uiteindelijke systeem zal bestaan uit een combinatie van een snelle en een optimale methode. De Gradient methode geeft snel een suboptimale oplossing die gebruikt kan worden, zolang een langzamere methode zoals een veiling of *Distributed Breakout* nog geen oplossing gevonden heeft.

Bibliografie

-J.A.A.J. Janssen, M.G. Maris, *Self-configurable distributed control networks on naval ships,* in SCSS 2003, Orlando, FL, 2003

-B. Demsky, M. Rinard, *Automatic Data Structure Repair for Self-Healing Systems,* First Workshop on Algorithms and Architectures for Self-Amnaging Systems. June 2003

-Shen, W.-M., B. Salemi, and P. Will, *Hormone-Inspired Adaptive Communication and Distributed Control for CONRO Self-Reconfigurable Robots,* IEEE Transactions on Robotics and Automation, October 2002

-S. Russel, P. Norvig, *Artificial Intelligence a modern approach,* Prentice Hall International, Inc. 1995

-*Distributed Detection for Smart Sensor Networks,* http://w3.antd.nist.gov/wctg/smartsensors/sensornetworks.html

-M.G. Corr, C.M. Okino, *Networking reconfigurable smart sensors*, Thayer school of Engineering Dartmouth College Hanover NH 03755

-Fredrik Ygge, Hans Akkermans, *Power Load management as a Computational Market,* Proceedings of the 2nd International Conference on multi-agent systems ICMAS'96, Kyoto, Japan, December 1996

-Paul Havinga, The EYES Project: Energy Efficient Sensor Networks, http://eyes.eu.org/

-L.P. Claire, G.J. Pottie, J.R. Agre, *Self Organizing Distributed Sensor Networks*, Rockwell Science Center Thousand Oaks CA 91360 USA

- Lakshminarayanan Subramanian and Randy H.Katz, *An Architecture for Building Self-Configurable Systems,* Department of Elektrical Engineering and Computer Sciences, U.C. Berkeley

- Luc Hamilton, *Distributed Intelligence in CRitical Infrastructures for Sustainable Power,* http://www.ecn.nl/crisp/, Energy research Centre of the Netherlands

-D. Cockburn and N. R. Jennings, *ARCHON: A Distributed Artificial Intelligence System for Industrial Applications,* in Foundations of Distributed Artificial Intelligence (eds. G. M. P. O'Hare and N. R. Jennings) Wiley, 1996, 319-344.

-T. Sandholm, *Distributed Rational Decision Making. In the textbook Multiagent Systems: A Modern Introduction to Distributed Artificial Intelligence*, Weiß, G., ed., MIT Press. Pp. 201-258, 1999

-B. Huberman, S.H. Clearwater, *A multi-agent system for controlling building environments,* ICMAS-95, pages 171-176, San Fransisco, CA, june 1995