FACULTY OF ENGINEERING AND ARCHITECTURE

Taming a swarm of robot locusts: how scale-free networks emerge from local interactions

Timothy Thiecke
Student number: 01004148

Supervisors: Prof. dr. ir. Pieter Simoens, Dr. Yara Khaluf
Counsellors: Dr. Yara Khaluf, Ilja Rausch, Johannes Nauta

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Information Engineering Technology

Academic year 2019-2020

GHENT UNIVERSITY

Ontstaan van schaalvrije netwerken door lokale interacties in een robotzwerm: een gevallenstudie met robot sprinkhanen

Timothy Thiecke
Promotors: Prof. dr. ir. Pieter Simoens, dr. Yara Khaluf
Begeleiders: Dr. Yara Khaluf, Ilja Rausch, Johannes Nauta

***Abstract*—Het veld van kunstmatige intelligentie gebruikt collectieve systemen om problemen op te lossen. Individuen maken beslissingen op basis van de staat van de lokale omgeving. Een bestaand experiment over het gedrag van sprinkhanen in zwermen heeft vastgesteld dan een schaal-vrij netwerk onstaat op basis van de lokale interacties van dezelfde individuen. De masterproef tracht een antwoord te geven op de onduidelijkheid waarom dit globale complex gedrag ontstaat.**

I. Introductie

In het veld van kunstmatige intelligentie wordt gebruik gemaakt van gedistribueerde collectieve systemen om problemen op te lossen. *Agents* (individuen) maken beslissingen op basis van informatie gekregen op de lokale schaal waardoor op de globale schaal complex gedrag kan ontstaan. Wanneer het systeem collectief een beslissing kan nemen is dit systeem gesynchroniseerd. Op basis van een experiment omtrent het modelleren van het gedrag van sprinkhanen in zwermen is vastgesteld dat een schaalvrij netwerk ontstaat door de lokale interacties van het systeem. Het is tot nu toe niet goed begrepen waardoor dit complex gedrag ontstaat. Schaalvrije netwerken zijn van interesse voor de wetenschappelijke gemeenschap omdat ze indicatief zijn van complexe subsystemen. Tevens komen distributies vaak voor in de natuur in verschillende situaties [1]. De thesis pakt het probleem aan door te trachten een antwoord te geven op het probleem van de onduidelijkheid omtrent het ontstaan van een schaalvrij netwerk.

A. Overzicht van de paper

Sectie II legt meer in detail uit wat een schaalvrij netwerk inhoudt. Sectie III beschrijft de opstelling van het experiment en legt uit hoe het netwerk wordt opgebouwd. Sectie IV bespreekt hoe het schaalvrije netwerk kan vastgesteld worden aan de hand van data gegenereerd tijdens het uitvoeren van het experiment. Sectie V presenteert empirische observaties en hoe ze kwantitatief kunnen geëvalueerd worden. Sectie VI bouwt voort op de bevindingen van sectie V.

II. Wat is een schaalvrij netwerk?

In deze sectie word kort uitgelegd wat het concept van een schaalvrij netwerk is, aangezien kennis van dit onderdeel van belang is voor de discussie van de volgende secties.

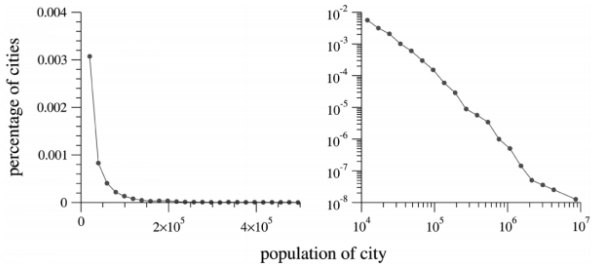


Fig. 1: Links: de kansverdeling van bevolkingsaantallen van steden in de VS; Rechts: de linker histogram op logaritmische schaal [3]

Schaalvrij impliceert dat een of andere metriek onafhankelijk is van de schaal van het systeem waar het zich in bevindt. Deze metriek fungeert hetzelfde ongeacht de schaal. Voor het veld van *Swarm Intelligence* zijn schaalvrije eigenschappen gewenst aangezien men vaak met systemen werkt met een groot aantal individuen.

Een schaalvrij netwerk is een netwerk (of graaf) waarvan de graden van de knopen een machtsrelatie volgen. Een machtsrelatie is een evenredig verband tussen twee variabelen aan de hand van een macht. Een algemeen voorbeeld is het berekenen van de oppervlakte van een vierkant, waarbij de oppervlakte kwadratisch toeneemt bij een lineaire toename van de zijde. Stel dat k de graad aangeeft van een willekeurige knoop in de graaf. Indien de kans dat de knopen in de graaf met graad k bepaald wordt door een machtsrelatie in de vorm van $p(k) = k^{-\alpha}$, dan volgt de distributie van graden in het netwerk een machtsrelatie. Deze stelling geldt in beide richtingen [2].

Bij het nemen van de logaritme van deze vergelijking volgt vergelijking $\ln p(k) = -\alpha \ln k$. Dit suggereert dat er op de logaritmische schaal een lijn kan geobserveerd worden. Een populair voorbeeld uit de literatuur toont aan dat de bevolkingsaantallen in de Verenigde Staten op analoge wijze verdeeld zijn zoals te zien is in figuur 1. Een lijn op de logaritmische schaal is een noodzakelijke maar onvoldoende voorwaarde om te kunnen concluderen dat een machtsrelatie in een bepaalde distributie voorkomt [1] [2].

De functie divergeert voor waarden van $k = 0$. Er is met andere woorden nood aan een waarde k_{min}. Het gedeelte

van de verdeling voor waarden van $k > k_{min}$ heet de machtsrelatie staart. In het geval van een netwerk moet de graad minstens 1 zijn. Knopen die niet verbonden zijn met de samenhangende componenten van de graaf worden niet meegeteld in de distributie van graden [1].

Een schaalvrije graaddistributie heeft als gevolg dat een groot aantal knopen in het netwerk een kleine graad heeft. Omgekeerd zijn er slechts een klein aantal knopen met een uitzonderlijk graad. Deze knopen worden gedefinieerd als *hubs* [2].

De thesis onderzoekt de factoren die leiden tot het ontstaan van schaalvrije netwerken in het experiment zoals beschreven in sectie III.

III. Experiment

In deze sectie wordt beschreven hoe het experiment is opgesteld en de interne dynamiek.

Het experiment is opgesteld in de *Autonomous Robots Go Swarming* (ARGoS) simulator[4] en geschreven in C++. ARGoS geeft de mogelijkheid om grote zwermen van robots te simuleren in een virtuele omgeving zonder de feitelijke materiële kosten ervoor te moeten betalen. De code die geschreven wordt voor de virtuele implementaties van de robots kan geïnstalleerd worden op hun fysieke tegenhangers.

Het oorspronkelijke experiment is opgesteld als implementatie van het onderzoek op het gedrag van sprinkhanen in zwermen die afhankelijk van de dichtheid van de zwerm uiteindelijk op een collectieve manier bewegen[5]. Het gedrag van de virtuele sprinkhanen werd uitgebreid met een beslissingsmodel dat op een dynamische wijze de communicatiestraal kan aanpassen aan de hand van de staat van de lokale omgeving[6]. Tevens werd ruis geïntroduceerd door op elk moment de mogelijkheid aan de robots te geven om te veranderen van richting [7].

De robots van het experiment hebben onder andere de mogelijkheid om door de wereld te bewegen alsook andere robots te detecteren aan de hand van een sensor-actuator koppel binnen een bepaalde communicatiestraal. De actuator van een robot zendt binnen de communicatiestraal uit dat de robot zich daar bevindt. Andere robots die zich binnen deze straal bevinden krijgen via ARGoS op hun sensor een bericht binnen dat een robot zich in hun omgeving bevindt. De manier hoe ARGoS communicatielinks opstelt is weergegeven in figuur 2. Belangrijk hierbij is dat robots met een grote communicatiestraal waarin meerdere robots zich bevinden toch kunnen denken dat ze zich alleen bevinden in de omgeving begrensd door de communicatiestraal.

Het beslissingsmodel bekijkt de staat van de lokale omgeving en beslist vervolgens of de communicatiestraal wordt vergroot of verkleind (of geen van beiden), afhankelijk van de synchronisatie staat van de omgeving. De synchronisatie metriek wordt bepaald door het absolute verschil tussen de fractie van buren die naar links gaan min de buren die naar rechts gaan. Indien er geen buren in de omgeving worden gedetecteerd, dan kijkt de robot of de omgeving de staat van *breakdown* heeft bereikt. *Breakdown* gebeurt wanneer de verandering van de omgeving groter is dan die van de vorige simulatieframe. Als dit het geval is, dan wordt de communicatiestraal ook vergroot. De beslissing wordt daardoor genomen op de lokale schaal. Een communicatielink wordt beschouwd als een ongerichte verbinding van een graaf die wordt bijgehouden op de globale schaal. Tevens wordt elke robot beschouwd als een knoop in dezelfde graaf. Het blijkt dat in vele gevallen dit globale netwerk schaalvrij is en dat door lokale interacties een complexe globale eigenschap ontstaat. In figuur 3 wordt weergegeven hoe het experiment er als gevolg uitziet.

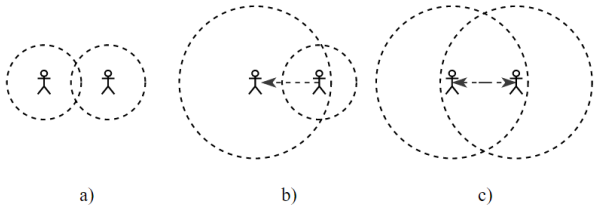


Fig. 2: De verschilende communicatiestaten tussen robots. Het mens icoon representeert een robot, de gestippelde cirkel definieert de lokale omgeving begrensd door de communicatiestraal en de pijlen tussen mensen weergeven de gerichte communicatielinks a) Alhoewel er overlap is tussen de communicatiestralen is de afstand tussen twee te groot voor een communicatielink b) De linker robot heeft een grotere communicatiestraal. De rechter robot bevindt zich in de omgeving van de linker robot waardoor ARGoS bepaalt dat er een communicatielink is van rechts naar links c) Beide robots kunnen elkaar detecteren. ARGoS bepaalt dat er twee communicatielinks zijn (in beide richtingen 1)

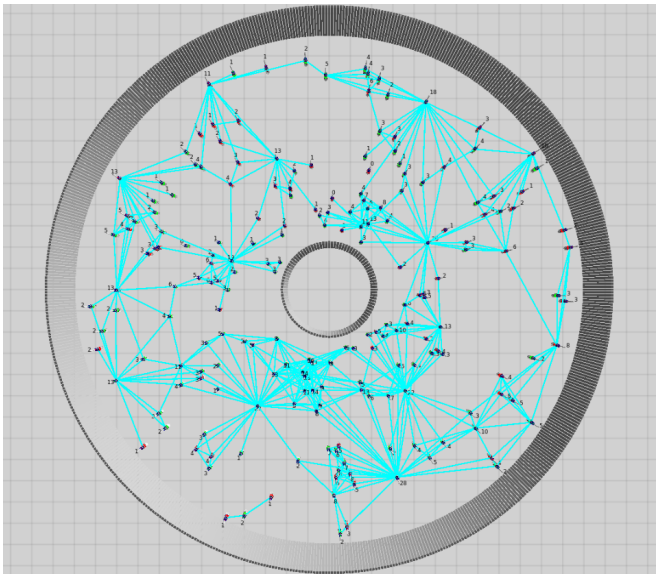


Fig. 3: Een screenshot van de virtuele wereld van het experiment. Elke zwarte stip representeert een robot. De blauwe lijen tussen de robots representeren de (gerichte) communicatielinks

IV. Observeren van de schaalvrije distributie

In deze sectie wordt uitgelegd welke distributies worden gegenereerd door het experiment en hoe ze kunnen geanalyseerd worden.

De graaddistributie van het netwerk moet geëvalueerd worden om aan te kunnen tonen dat het een machtsrelatie volgt. Als er kan vastgesteld worden dat de distributie een machtsrelatie volgt dan is het netwerk schaalvrij. Deze stelling geldt in beide richtingen.

Tijdens het verloop van het experiment worden twee graaddistributies gegenereerd: de κ- en λ-distributies. De κ-distributie is equivalent aan de graaddistributie aan het einde van het experiment. Deze distributie kan sterk verschillen met zijn voorganger of opvolger. De λ-distributie is de som van de gerangschikte graaddistributies per simulatieframe waarvan het gemiddelde wordt genomen aan het einde van het experiment. In beide gevallen worden de distributies gesorteerd van laag naar groot.

Beweren dat een bepaalde distributie een machtsrelatie volgt kan als controversieel beschouwd worden zonder verder grondig onderzoek. Een algemene methodologie bepaalt schatters voor de waarden van α en k_{min}. De waardes voor deze variabelen vormen de machtsrelatie hypothese. Een probleem is dat zo een statistische fitting in feite kan toegepast worden op elke mogelijke distributie. De machtsrelatie hypothese wordt geëvalueerd aan de hand van een *goodness-of-fit* test die een p-waarde, de metriek voor de plausibiliteit van de hypothese, retourneert. Als deze waarde onder een bepaalde drempel valt, dan wordt de hypothese afgekeurd, anders wordt ze als plausibel beschouwd. Om de hypothese nog meer te maatstaven wordt in de laatste stap van de methodologie een *log-likelihood ratio* test uitgevoerd. Het teken van deze ratio kan aangeven welke fit beter past bij de oorspronkelijke distributie. Tevens wordt hier ook een andere p-waarde gegenereerd die aangeeft of het teken van de ratio doorslaggevend genoeg is. Ook hier wordt gebruik gemaakt van een drempelwaarde. Echter, als de waarde groter is dan de drempel dan zijn beide fits niet goed genoeg om toegepast te worden op de distributie [1].

In de context van dit experiment wordt gebruik gemaakt van een combinatie van twee bestaande implementaties om te bepalen of de graaddistributie schaalvrij is [8][9].

V. Heterogene gedragsmatige rollen

Op basis van empirische observaties werd vastgesteld dat alhoewel het experiment begint met initiële homogene condities, dat er snel heterogeen gedrag ontstond. Het werd duidelijk dat er verschillende vormen van gedrag konden ingedeeld worden in rollen. Deze sectie beschrijft in het kort de gevonden rollen alsook een kwantitatieve analyse er van.

In eerste instantie werd de rol van *isolated hub* vastgesteld. Dit zijn knopen die initieel geïsoleerd van de rest van de knopen starten. Isolatie kan kwantitatief bepaald worden door het berekenen van de afstand van een knoop tot zijn dichtstbijzijnde buur. Hoe groter deze metriek ten opzichte van andere metingen, des te groter de kans dat de robot geïsoleerd is. Doordat de robot consequent geïsoleerd is, kan de communicatiestraal van de robot enkel vergroot worden wanneer de robot de *breakdown* staat bereikt. Deze robot krijgt hierdoor een grote communicatiestraal maar detecteert in feite geen andere robots waardoor dat er een positieve feedback lus ontstaat. Hierdoor hebben *Isolated hubs* een uitzonderlijk grote graad op de globale schaal.

Vervolgens werd gekeken naar robots die in geclusterde omgevingen beginnen. Aangezien deze robots wel andere robots detecteren zal het beslissingsmodel bepalen of de lokale omgeving op een synchrone manier beweegt. Robots in synchrone omgevingen hebben de tendens om hun communicatiestraal te verkleinen als gevolg van de beslissing van het beslissingsmodel. Omgekeerd, zijn er robots die zich in asynchrone omgevingen bevinden met als gevolg dat ze hun communicatiestraal vergroten. Robots in asynchrone omgevingen kunnen in asynchrone omgevingen blijven bestaan omdat de dynamische omgeving er toe leidt dat er steeds nieuwe robots in de buurt er van komen en oude robots verdwijnen.

Om deze rollen kwantitatief te kunnen analyseren werd gebruik gemaakt van twee heuristieken om representatoren van de groepering te kunnen vinden. De *Nearest Neighbour Distance Heuristic* (NNDH) maakt gebruik van de initiële afstand tussen robots om ze te rangschikken en er vervolgens drie uit te selecteren. De kleinste, middelste en grootste elementen, respetievelijk n_{min}, n_{mid} en n_{max}, worden geselecteerd aan de hand van de sortering op basis van de heuristische metriek. Daarentegen maakt de *Post Degree Heuristic* (PDH) gebruik van de kennis op het einde van het experiment. De collectie van robots wordt gerangschikt aan de hand van de graad van de knoop op het einde van het experiment. Analoog met de eerste heuristiek worden n_{min}, n_{mid} en n_{max} gekozen en gemarkeerd. De gemarkeerde knopen houden verschillende vormen van geschiedenis data bij doorheen het experiment en wordt op het einde in bestanden geplaatst voor verdere analyse.

Figuren 4, 5 en 6 geven een overzicht van de ontwikkeling van geschiedenis data door de n_{max} knoop geselecteerd door de NNDH. Het valt op dat deze knoop een sterke stijging van zijn globale graad en communicatiestraal ervaart. Dit is te wijten aan het herhaaldelijk bereiken van de *breakdown* staat omdat deze knoop geïsoleerd is in de virtuele omgeving. Omdat zijn eigen communicatiestraal groot is, wordt de knoop ontdekt door zijn buren in de lokale omgeving en wordt de gerichte communicatielink getransformeerd op de globale schaal.

Een analoog verloop wordt weergegeven in figuren 7, 8 en 9. Hierbij gaat het echter om de n_{min} knoop geselecteerd door de PDH. Het valt op dat deze knoop een lage graad heeft doorheen het experiment, alsook dat de knoop begint in een geclusterde regio. Aangezien de synchronisatie metriek van de regio initieel hoog ligt, wordt de communicatiestraal van de knoop verkleind. In het midden van het verloop van het experiment is dit echter niet meer vanzelfsprekend omdat de omgeving asynchroon is geworden, waardoor de communicatiestraal weer vergroot.

Voor beide heuristieken is de knoop n_{max} een sterke heuristiek om een *isolated hub* te vinden. De knopen n_{min} en n_{mid} zijn vooral interessant om het verloop van knopen in geclusterde omgevingen te observeren, en meer specifiek

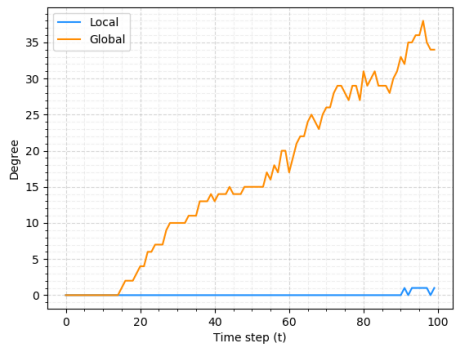


Fig. 4: De ontwikkeling van de graden van de n_{max} knoop geselecteerd via NNDH ($t_{end} = 100$,seed=10,N=200)

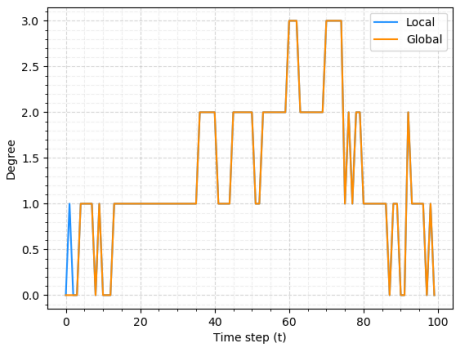


Fig. 7: De ontwikkeling van de graden van de n_{min} knoop geselecteerd via NNDH ($t_{end} = 100$,seed=10,N=200)

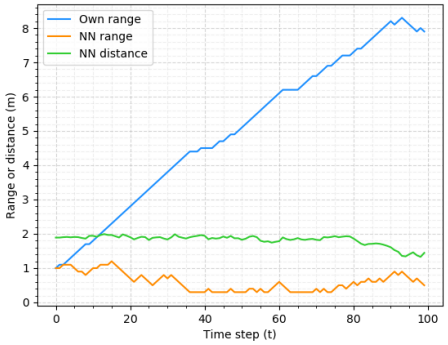


Fig. 5: De ontwikkeling van de communicatiestraal van de n_{max} knoop geselecteerd via NNDH in relatie met zijn dichtstbijzijnde buur ($t_{end} = 100$,seed=10,N=200)

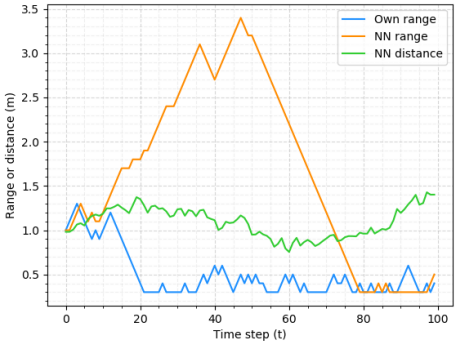


Fig. 8: De ontwikkeling van de communicatiestraal van de n_{min} knoop geselecteerd via PDH in relatie met zijn dichtstbijzijnde buur ($t_{end} = 100$,seed=10,N=200)

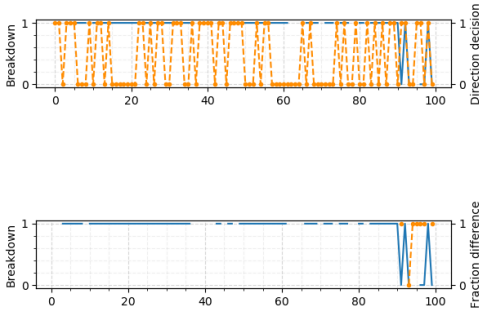


Fig. 6: De ontwikkeling van *breakdown* detectie, de richtingsbeslissingen en de synchronisatie metriek van de lokale omgeving van de n_{max} knoop geselecteerd via NNDH ($t_{end} = 100$,seed=10,N=200)

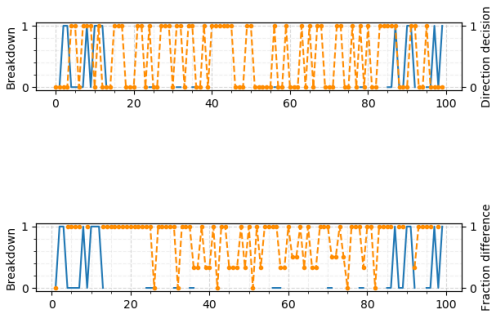


Fig. 9: De ontwikkeling van *breakdown* detectie, de richtingsbeslissingen en de synchronisatie metriek van de lokale omgeving van de n_{min} knoop geselecteerd via PDH ($t_{end} = 100$,seed=10,N=200)

het gedrag in context van de mate van een synchrone of asynchrone omgeving.

Het is duidelijk uit de analyse dat de rol van een robot niet vastligt voor de volledige lengte van het experiment en er een *fuzzy* overgang is tussen de verschillende rollen.

VI. Analyse van het beslissingsmodel

De bevindingen gevonden via de observaties en heuristieken uit sectie V leiden tot een korte bespreking van het belang van het beslissingsmodel. Er is duidelijk een intieme relatie tussen de communicatiestraal van een robot en zijn graad in de globale graaf.

Om het effect van het beslissingsmodel te testen werden er in de code nieuwe checks ingevoerd die de *breakdown* of de lokale omgeving controle aan en uit kunnen zetten. Hierdoor ontstaan 4 verschillende staten: het oorspronkelijke beslissingsmodel, het beslissingsmodel met enkel de *breakdown* controle, het beslissingsmodel met enkel de lokale omgeving controle en ten slotte geen beslissingsmodel. Vervolgens werd het experiment herhaaldelijk uitgevoerd over 30 seeds. De κ- en λ-distributies werden gesommeerd over de seeds en het gemiddelde eruit genomen met als doel de statistische fluctuaties te verminderen.

Tabel I toont dat het volledige beslissingsmodel met een hoge kans zorgt voor het ontstaan van een schaalvrij netwerk. Het beslissingsmodel met enkel de *breakdown* controle zorgt ook voor een relatief betrouwbaar succes. De beide andere staten verminderen de kans op het succes van het ontstaan van een schaalvrij netwerk.

VII. Conclusie

Het doel van de masterproef was een antwoord proberen vinden op het ontstaan van een schaalvrij netwerk louter door interacties op het lokale niveau. Er werd vastgesteld dat het beslissingsmodel verantwoordelijk is voor een heterogene wereld die ontstaat uit homogene begincondities. Hierdoor ontstaan verschillende gedragsmatige rollen die kunnen gevonden worden aan de hand van verschillende heuristieken. Deze rollen liggen niet vast voor de gehele duur van het experiment.

Volgend onderzoek moet proberen het beslissingsmodel verder te abstraheren om hetzelfde gedrag al dan niet het ontstaan van een schaalvrij netwerk te kunnen reproduceren in een andere context.

References

[1] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," 2007.

[2] A.-L. Barabási and M. Pósfai, *Network science*. Cambridge: Cambridge University Press, 2016. [Online]. Available: http://barabasi.com/networksciencebook/

[3] M. Newman, "Power laws, pareto distributions and zipf's law," *Contemporary Physics*, vol. 46, no. 5, p. 323–351, Sep 2005. [Online]. Available: http://dx.doi.org/10.1080/00107510500052444

[4] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[5] J. Buhl, D. J. T. Sumpter, I. D. Couzin, J. J. Hale, E. Despland, E. R. Miller, and S. J. Simpson, "From disorder to order in marching locusts," *Science*, vol. 312, no. 5778, pp. 1402–1406, 2006. [Online]. Available: https://science.sciencemag.org/content/312/5778/1402

[6] I. Rausch, A. Reina, P. Simoens, and Y. Khaluf, "Coherent collective behaviour emerging from decentralised balancing of social feedback and noise," *Swarm Intelligence*, vol. 13, no. 3, pp. 321–345, Dec 2019. [Online]. Available: https://doi.org/10.1007/s11721-019-00173-y

[7] C. Huepe, G. Zschaler, A.-L. Do, and T. Gross, "Adaptive-network models of swarm dynamics," *New Journal of Physics*, vol. 13, no. 7, p. 073022, Jul 2011. [Online]. Available: http://dx.doi.org/10.1088/1367-2630/13/7/073022

[8] J. Alstott, E. Bullmore, and D. Plenz, "powerlaw: A python package for analysis of heavy-tailed distributions," *PLoS ONE*, vol. 9, no. 1, p. e85777, Jan 2014. [Online]. Available: http://dx.doi.org/10.1371/journal.pone.0085777

[9] T. Nepusz, "plfit: Fitting power-law distributions to empirical data," https://github.com/ntamas/plfit, 2019.

TABLE I: Kans dat een schaalvrij netwerk (SVN) ontstaat in relatie tot het aan en uit zetten van bepaalde onderdelen van het beslissingsmodel. De Aan en Uit kolommen geven de status aan van beide componenten (N=200, gemiddelde over 30 seeds)

	Uit	Lokale omgeving controle	*Breakdown* controle	Aan
Kans op SVN (κ)	0.2	0.067	0.2	0.67
Kans op SVN (λ)	0.0	0.0	0.7	1.0
Gemiddelde graad	1.78	0.32	5.63	6.89
Gemiddelde communicatiestraal	1.0	0.5	1.41	1.43

Contents

List of Figures

List of Tables

List of Listings

1

Introduction

The field of Artificial Intelligence uses distributed collective systems in order to solve problems. In such a system, the agents make decisions based on locally acquired information. A swarm making a collective decision is synchronised. Collective decision making has been attributed to naturally occurring phenomena, such as swarming of locusts. It has been established that swarms of locusts communicate on a local scale in order for complex behaviour to emerge on the global scale. Further analysis of this behaviour in a virtual environment of locusts showed that an additional layer of abstraction can be deduced, in which the local interactions can cause a global scale-free network to emerge.

Scale-free networks are of interest to the scientific community as they can be attributed to many distributions found in nature that cannot be fitted by more common distributions such as the normal distribution. Furthermore, they are indicative of complex subsystems and can yield new insights. Additionally, their scale invariant property is of use to collective systems as they behave qualitatively similar no matter what the size of the group is. From the mathematical point of view, the degree distribution of a scale-free network follows a power law.

Currently, it is not well understood why through mere local interactions complex behaviour can emerge, such as a scale-free network. The thesis aims to tackle this question through the following steps. First, an analysis of the emergence of scale-free properties in a simulated environment of swarming locusts is performed. Second, it is determined which mechanisms and parameters are

inherently responsible for this emergence. Third, there is need for a definition of the different types of agent behaviour emerging as a result of local interactions and how they affect the global state of the environment.

Chapter 2 introduces the reader to concepts, such as power law distributions and scale-free networks, in rising order of complexity required to understand the remainder of the thesis. Chapter 3 discusses the environment and setup of the experiment. Furthermore, it presents a generic methodology to fit power law distributions to data sets. Chapter 4 describes the emerging behavioural roles in the swarm, hypothesised and deduced through observations across numerous simulations of the experiment along with a discussion about empirical data. Furthermore, it discusses how the scale-free network is observed. Next, an analysis is offered of a core component of the simulation. The final chapters discuss the experiment from a different point of view.

2

Theoretical background

In the following sections, a few theoretical concepts are introduced in rising order of complexity. The reader is guided through and will obtain a basic understanding of the relevant notions that reoccur throughout the dissertation.

2.1 What is a network?

In its simplest form, a network or graph, is a collection of nodes (or vertices) which may be connected by one or multiple edges. Both nodes and edges may contain some form of information. A network is bounded by the amount of nodes N; $N-1$ forms the upper bound of the amount of edges M that can exist in an acyclic network.

An edge may have a direction from a certain node to another. If two nodes are connected by two edges in opposing directions, then those edges may be considered as one undirected edge. This property of edges is mutually exclusive in most cases of literature and in this thesis: either all edges of the network are directed or undirected.

A node has a degree equal to the amount of edges connected to it in an undirected network; in a directed network this number is separated in in-degree equal to the amount of edges that target this node as its end point and vice versa for out-degree.

A trivial example of a network is a social network where each node represents a person. Every type of relationship between various people may be represented by an edge, which may in turn contain some type of information about this relation.

2.2 What is a power law?

In the real world, many quantitative empirical observations can be fitted to a normal distribution where a concrete mean and standard deviation can easily be deduced. Examples from literature include heights of males or the speeds of cars on the highway [1]. Normal distributions are well known for their bell shaped curve with a well defined visual peak signifying the mean value which most other values revolve around.

A normal distribution loses its value when a mean is not easily deduced from these observations. Suppose that outliers systematically appear far from the mean, another distribution is then required to account for these observations during the data fitting procedure. A well known example discussed in the literature uses the population of US cities [1]. The US contains an abundance of cities of a small population size, compared to the existence of few mega-sized cities such as New York or Los Angeles. Plotting this data set to a histogram or probability density function (PDF) yields the left graph of figure 2.1. This in itself visually resembles log-normal or exponential distributions. However, note that the PDF diverges as x approaches zero which is not the case for these other distributions.

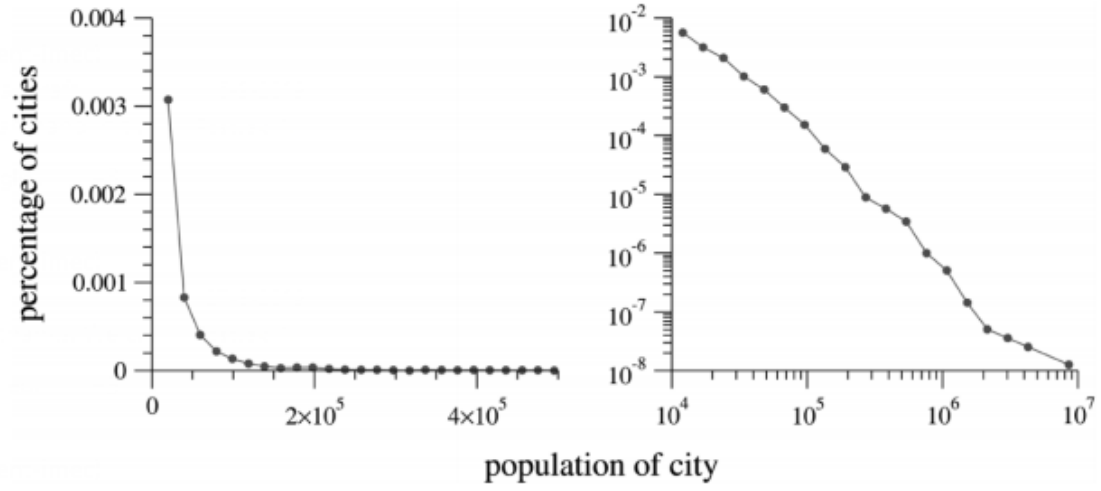


Figure 2.1: Left: the probability density function of population sizes in America; Right: the log-log plot of the left histogram [1]

Consider the right histogram of figure 2.1 which is the log-log plot of the original histogram.

What can be seen is a somewhat straight line appearing. According to Newman [1], since the log-log plot shows a straight line, this line can be approximated by a linear function of the form $y = ax + b$. y can be substituted with $\ln p(x)$ and ax with $-\alpha \ln x$. The negative sign is due to the slope of the line. After reversing the logarithmic scaling, equation 2.1 is obtained. α is known as the exponent of the power law and is constant. Note that it has a negative sign resulting in the descending line on the log-log plot. The constant C is the normalising constant to make sure that the PDF has a total probability of one. A distribution which produces output following equation 2.1 is called a power law.

$$p(x) = Cx^{-\alpha} \tag{2.1}$$

As noted before and backed by equation 2.1, $p(x)$ diverges as x approaches zero. Therefore, there is a need for some value of x_{min} from which $p(x)$ does not diverge. All points of x larger than x_{min} follow a power law. The distribution from x_{min} onward is called the power law tail [1].

Section 3.3 discusses more in depth the controversial nature of claiming that a distribution follows a power law, as one can never be certain that it does. It also offers a strategy to determine the exponent α and the value of x_{min} from an empirical observation data set, and how certain someone can be that such a data set may follow a power law or not.

2.3 What is a scale-free network?

A scale-free network combines the concepts discussed in the previous sections. Let k equal the degree of a node. If the fraction of nodes with degree k follows a power law, the complex network is said to be scale-free. This statement is valid in both directions [4]. Taking equation 2.1 and applying this rule results in equation 2.2

$$p(k) = k^{-\alpha} \tag{2.2}$$

Equation 2.2 is defined for undirected networks. However, in the case of directed networks, a separation of the distributions is necessary based on the in-degree and out-degree of a node. Thus, with in-degree k_{in} and out-degree k_{out} equation 2.2 is adopted to equations 2.3 and 2.4 [5].

$$p(k_{in}) = k^{-\alpha_{in}} \tag{2.3}$$

$$p(k_{out}) = k^{-\alpha_{out}} \tag{2.4}$$

A scale-free network has several characteristics that result from following a power law. They contain hubs: a few nodes with a significantly higher degree than the rest of the nodes in the network. Hubs appear in the tail section of the distribution and thus comprise a very small fraction of the network. Comparatively, there is an abundance of nodes that share few links with others [5]. This characteristic stems from the given distribution. It is in crucial contrast to random networks in which the plot of the degree distribution resembles a bell shape. Bonabeau and Barabási summarise it nicely [6]:

> "Hubs are simply forbidden in random networks."

Another characteristic of a scale-free network is its resiliency against random failures. Picking a node at random from the network has a low probability of destroying the scale-free property of the network. Conversely, a targeted attack on a hub can have disastrous effects. Bonabeau and Barabási [6] compare the effects of random and targeted attacks on both random and scale-free networks by painting an example of the US road network vs. US airspace. A targeted attack on a hub can cause the network to lose its connectedness. Random failures only have a small chance of affecting hubs. In the same work, Bonabeau and Barabási [6] mention that due to following a power law, the odds of choosing a random node connected to k other nodes is proportional to $1/k^n$.

Scale-free networks do not simply emerge at random; they can be attributed to complex subsystems. There is a need for a generative model that allows one to create a scale-free network. A common generative model combines two simple mechanisms: a growth process and a form of preferential attachment. A growth process begins from a few existing nodes in the network and adds a new node at every time-step to the existing nodes in the system. Preferential attachment is used when adding and linking to existing nodes; the probability of a node being linked to existing nodes is dependent of the degree of an existing node. The higher the degree, the higher the probability of getting linked [7]. Section 2.4 discusses the most popular model in more depth.

2.4 What is the Barabási-Albert model?

The Barabási-Albert model (BA model) is one of the first proposed generative models of scale-free networks. It offers a recipe for creating scale-free networks using two straightforward principles. The BA model differs from other models (Erdős' and Rényi's random graph or Watts' and Strogatz' small world models) for generating complex networks in that the emergence of

hubs is to be expected. The other models mention that the probability of finding a hub has an exponential falloff as k increases [8].

The first pillar of the BA model revolves around the growth process. It assumes (potentially) limitless growth of the network, as many complex systems in the real world also tend to grow for a long amount of time. The building process of a scale-free network starts with creating a number of nodes m_0. At every discrete time-step, a new node with an amount of edges m less than the initial network size m_0 is introduced and linked to the existing nodes [8].

The second pillar affects the way linking is done as described in the growth process. The nodes to which the newly created node will be linked is determined by linear preferential attachment. In simple terms, this constitutes a weighted random selection as the probability Π of a link occurring between nodes is determined by the degree k_i of the existing node i in relation to the sum of all the degrees in the network. Equation 2.5 shows this relation. Using these two principles will yield a scale-free network [8]. Nodes that are already strongly connected will have a higher probability of being linked to, leading to the rich-get-richer phenomenon which creates a positive feedback loop for the degrees of the highest connected nodes in the network. Therefore, the emergence of hubs is a natural result of the BA model.

$$\Pi(k_i) = \frac{k_i}{\Sigma_j k_j} \tag{2.5}$$

Both elements of the generative recipe are necessary, omitting one or the other has different effects. Removing preferential attachment yields an exponential distribution of the degrees rather than the desired scale-free one. Whilst removing the growth process may result in an early scale-free network emerging but will eventually result in a fully connected network [7].

There are limitations to the BA model; it is not the end-all of generative models for scale-free networks. The exponent of the generated scale-free distribution lies between two and three, whereas power laws support any exponent larger than one [8].

3

Experiment setup

This chapter introduces the reader to the simulation environment designed on the discoveries from the previous chapter and the techniques used to discover the distribution of an emerging scale-free network from that environment. Furthermore, it sets the stage for chapter 4.

Section 3.1 gives a quick and general overview of the simulator used. Section 3.2 delves through the crucial code in order to simulate the behaviour. Section 3.3 briefly describes the way the data was produced by the simulator and is analysed in order to determine if a scale-free network has emerged or not.

3.1 ARGoS simulator

Autonomous Robots Go Swarming (ARGoS) is a swarm simulator written in C++. It serves as a starting point of simulating numerous robots which can be ported over from the virtual to the real world. ARGoS is built around two core design motivations: flexibility and efficiency. Flexibility is offered in the form of modularity of components, run-time plugins, and straightforward extensibility to existing components. Efficiency is achieved by allowing the real-time simulation of numerous robots. As the swarm grows, the time complexity of the simulation also grows linearly. It is an ideal tool to setup an environment to run an experiment in the same

logic as described in section 1 [2].

An overview of ARGoS' architecture can be found in figure 3.1. The white boxes are plug-ins that can be extended in one way or another and show how flexible the simulator is [2]. Flexibility is configuration driven: a configuration file (.argos) in XML format is supplied as a parameter to the program which loads plugins accordingly. The extensions to the Controller and Loop functions plug-ins are described more in depth in subsections 3.2.2 and 3.2.3 respectively, as they form the backbone of the experiment itself. Other plug-ins use their default implementation as supplied by ARGoS.

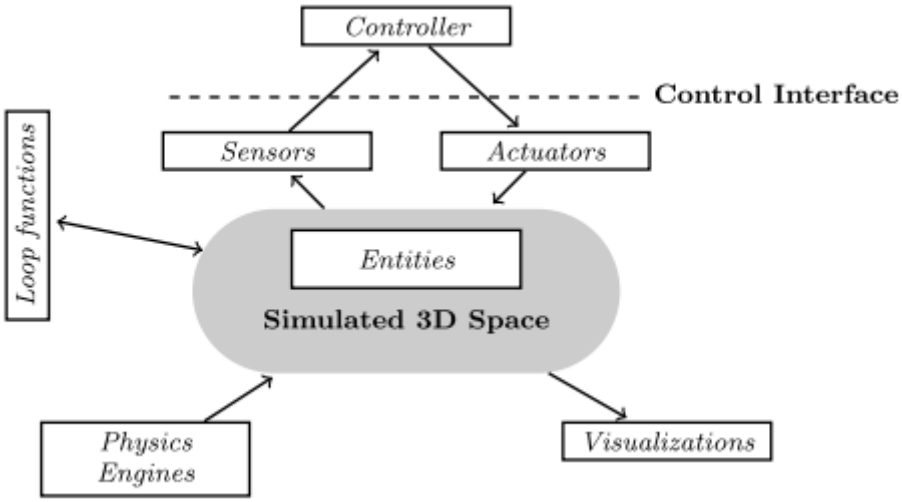


Figure 3.1: The architecture of ARGoS [2]

ARGoS is a deterministic simulation engine; running the same experiment with the same random seed and parameters will always yield the same results. This applies to the distribution of robots in the virtual world, the decisions made, and the numeric ID of a robot.

3.1.1 The marching foot-bot

The foot-bot is the unit of the swarm. Its implementation is based on the marXbot from the work of Bonani et al [9]. It offers logic such as basic movement, obstacle avoidance, and light sensing among others, though this dissertation will lay its focus on the robot's communication capabilities.

Robots can communicate with one another through their range-and-bearing (RAB) sensor-actuator pair. As dictated by the architecture, the communication is divided into sensor and

actuator implementations[1]. The communication framework's implementation is based on the work as proposed by Roberts et al [10].

Communication is message based. The sensor detects messages coming from the actuator of another robot. In the same way, the robot can broadcast messages to other robots through its actuator [2]. This communication exchange is not unbounded; each robot has a range variable. Connections between robots can be made between other robots that are within range of one another. Subsection 3.1.2 discusses the way ARGoS determines communication links between robots more in-depth and the problems that stem from it.

The default implementation of ARGoS for the foot-bot is not enough. The plug-in is extended with a decision making model which affects the communication range at each discrete time-step. The discussion of this extension and the decision making model can be found in section 3.2.2. From this point forward, expanding on the definition of a network from section 2.1: a node is equivalent to a robot (or foot-bot), and an edge is equivalent to a communication link between two robots.

3.1.2 Communication links

As mentioned in the previous subsection, communication is limited by the range of a robot. The range of a robot is a parameter to its sensor and actuator. As multiple robots exist that each have a sensor-actuator pair, some mechanism is required to mediate them. This is achieved by the range-and-bearing medium, a resource object[2] which is updated during every simulation step. It contains references to every entity in the world equipped with a RAB sensor-actuator pair, calculates distances between robots, and finally determines which robots are in range of one another. Listing 1 describes through pseudo-code in a simplified way how this update is implemented [11]. The EuclidianDistance function has the option of evaluating occlusion; it will only yield a useful distance if the two robots can see each other without any obstacles such as geometry or other robots. The RoutingTable is implemented as a mapping between entity ID and a set of pointers to the instances of other entities in the world that the robot is able to see. After the update call, the RoutingTable is filled up. Each robot's sensor can then request from the medium the set of robots that it is able to communicate with, which in turn is used to fill

[1] This split is important, as other types of sensor-actuator combinations exist (e.g. light sensing, obstacle avoidance)

[2] Comparable to a singleton

up the sensor's readings.

Input: Empty RoutingTable

Result: The RoutingTable filled in with existing connections

```
foreach Robot ∈Robots do
    foreach OtherRobot ∈OtherRobots do
        Distance ←EuclidianDistance(Robot, OtherRobot);
        if Distance < OtherRobot.Range then
            add connection between Robot and OtherRobot to RoutingTable;
        end
        else if Distance < Robot.Range then
            add connection between OtherRobot and Robot to RoutingTable;
        end
    end
end
```

Listing 1: A simplification of the way ARGoS determines links between two robots. Note that a link is directed.

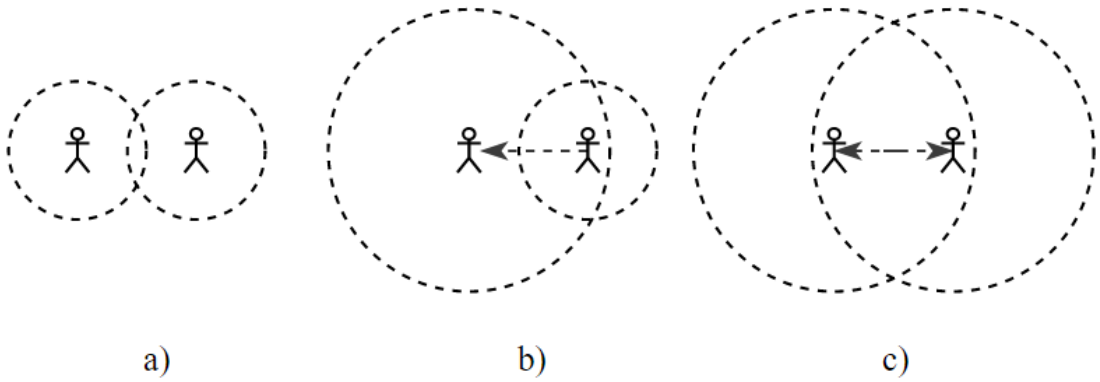


Figure 3.2: The different states that robots can be in relation to one another through their communication methods. The human icon represents a robot, the dotted lines the limits of their communication range a) Although the communication ranges have some overlap, the distance between the robots is too great and they do not constitute a connection. b) The left robot has a larger communication range, but a connection is added from the right robot to the left. c) Both robots share the same communication range and are able to detect each other. ARGoS adds a link in both directions

The way the ranges work in relation to one another is visualised in figure 3.2. In a way, ARGoS

implicitly builds a directed graph from local interactions when robots enter each others communication range. In figure 3.2, case c) is the default behaviour at the start of the simulation, when all of the robots are initialised with the same range parameter. However, in the experiment as described in this thesis, this will rarely be the case as all robots are dynamically able to change their own parameter. Therefore, figure 3.2' case b) occurs most often during the experiment. From this point forward two things should be considered: first, a robot that can communicate with another is defined as its neighbour. Second, the number of neighbours a node has can be considered as its local degree. Subsection 3.2.3 expands on these considerations.

3.2 Experiment code

ARGoS runs simulations in discrete time steps. In essence, the simulation step is comprised of three elements: the PreStep of the Loopfunction, numerous ControlSteps per robot, and the PostStep of the Loopfunction. A simplified overview is shown in figure 3.3.

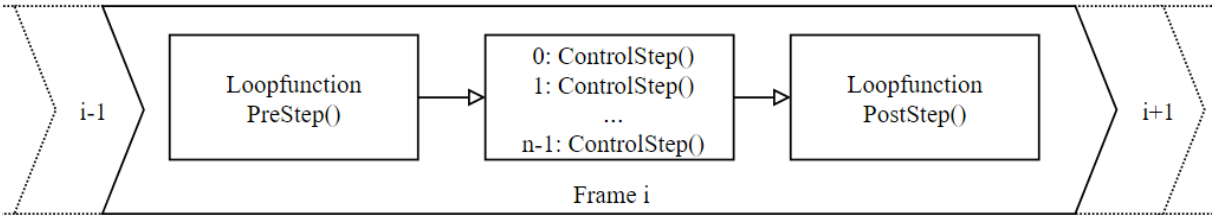


Figure 3.3: Simplified overview of a simulation step. Note that ControlStep is called for each robot in the world

The Controller of a robot is responsible for making decisions for the entity that represents the robot in the virtual world. A Controller exists per robot in the world and may affect the state of its robot. A Controller should only be able to make decisions based on information on a local scale. A Loopfunction is a resource object that can access all the information about the state of the world. It is allowed to access each robot's state, but, should not change it as that would interfere with the decentralised nature of the robot's decision making. Mainly, it is used to compute data which exists on a global scale.

Subsection 3.2.1 continues on the considerations mentioned at the end of 3.1.2. Subsection 3.2.2 discusses the extended ControlStep of the foot-bot. Subsection 3.2.3 discusses what happens in the global state.

3.2.1 Local vs. global state

The end of subsection 3.1.2 introduced the concept of a neighbour and the robot's degree count based on the number of neighbours. It is important to note that a separation between a local and a global degree is necessary.

Consider the local degree of a node to be equivalent to the neighbours that the sensor detects through the model of listing 1. Previously, it was mentioned that ARGoS creates a directed graph of local interactions where an edge indicates the direction of connection. When considering directed as undirected communications links, they can be unconditonally added to an undirected graph. Transforming the local ARGoS graph to an undirected global graph leads to changes in the degrees of a robot, referred to as its global degree.

However, a robot can only 'see' through its sensor readings and only knows its local scope. The local graph is constructed by ARGoS itself. Therefore, the Loopfunction can be used to track the global graph. It is the distribution of degrees in this global graph that turns out to be scale-free. Subsection 3.2.3 discusses the way to maintain this global graph.

3.2.2 Controller

A Controller is defined by the implementation of its ControlStep function, which is called once per robot each time step. The Controller in this experiment is an extension to the existing foot-bot implementation. The result of the ControlStep function is to determine how the communication range of a robot is adjusted. Next to the default behaviour of the foot-bot, the marching foot-bot has three pillars that cause the emerging behaviour to occur. The implementation of the marching foot-bot was carried out in the work of Rausch et al [12].

This subsection goes over the three pillars that define the decision making of a marching foot-bot. The emerging behaviour is discussed in chapter 4.

Average velocities and fractions

In the first step of the ControlStep, variables such as average velocity, and the fraction of robots going counterclockwise or clockwise are calculated to serve as inputs to the succeeding steps. They are dependent on the state of the node itself and the robot's local neighbourhood, in particular the neighbours' velocities. Its implementation is based on the Czirók model [13] which inspired the work surrounding collective marching in locusts by Buhl et al [14].

The work of Rausch et al [12] further reduces this to a binary problem where robots either go

left-counterclockwise, or right-clockwise based on the velocity of a robot. The sign of its velocity parameter determines the direction and helps simplify the way fractions of robots going either left or right are calculated. The sign affects the way the robots are represented visually, either having a green or a red light on them.

The variables are evaluated during each simulation frame. The average velocity is set as a parameter on the robot as the initial goal of this behaviour model is to create a local consensus of moving in the same direction [12]. The velocity of the robot is then used as a parameter of this step in the following simulation frame. Pseudo-code 2 offers a way to implement this evaluation. After this initial phase, the fraction variables are used as parameters to the next

steps.

Input: DegreeLocal
Output: FractionDifference, AverageVelocity
Result: Determines the difference of fractions and the average velocity of the robot
$Velocity$ ←0.0;
$AverageVelocity$←0.0;
$FractionLeft$←0.0;
$FractionRight$←0.0;
foreach *Neighbour* ∈*Neighbours* **do**
  $Velocity$ ←Neighbour.Velocity;
  if $Velocity > 0$ **then**
    increment FractionRight;
  else
    increment FractionLeft;
  end
  add $Velocity$ *to* $AverageVelocity$*;*
end
average out AverageVelocity;
if $AverageVelocity > 0$ **then**
  increment AverageVelocity;
else
  decrement AverageVelocity;
end
divide AverageVelocity in half;
Random ←RNG(−1.0, 1.0);
add Random to AverageVelocity;
set AverageVelocity to robot velocity;
if $AverageVelocity > 0$ **then**
  increment FractionRight;
else
  increment FractionLeft;
end
FractionLeft ←FractionLeft / $DegreeLocal + 1$;
FractionRight ←1.0 - FractionLeft ;
FractionDifference ←Abs(FractionLeft-FractionRight);

Listing 2: Fraction and average velocity algorithm. Note that RNG stands for random number generator

The calculation of the fraction difference δ and average velocities is dependent on the node itself and any detected neighbour in the populated neighbourhood. The velocity of every neighbour is evaluated, an average velocity is calculated and averaged, and either fraction is incremented. Some random fluctuations are added by generating and adding a random number to the average velocity and is then set as the velocity of the robot. δ is then calculated which serves as an important parameter for the next step in the decision process.

First, the decision making of these nodes is not influences by the breakdown detection. Their local degree is based on the readings that they have received from their sensor. As a result of this non-zero value, the section of the decision making model which deals with breakdown is ignored. Inside of the non-zero decision path, random numbers are generated twice and compared with δ. The robot may or may not increase or decrease its range based on that comparison as this decision is probabilistic. Due to the fact that these robots have a consistent small range, it can be deduced that one of two things happens for most of the time steps: either the robot decides to do nothing or it decides to decrease the range.

The deduction can be further explained by taking another closer look at the way the δ behaves. This is related to the way the fractions are highly correlated. δ is the absolute value of the subtraction of the two fractions, which can easily be written as equation 3.1 where x is either the left or the right fraction. Figure 3.4 offers a plot of this function. It becomes apparent that δ approaches its maximum if either fraction of robots approaches zero or one. Therefore, a high δ is indicative of a synchronised local neighbourhood.

$$\delta = f(x) = |2x - 1| \tag{3.1}$$

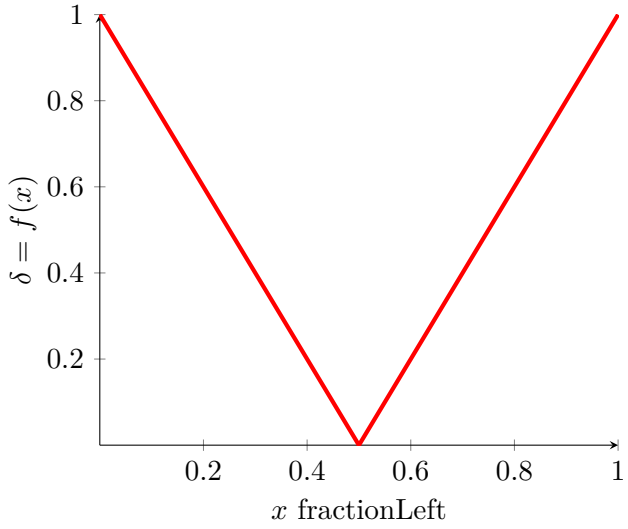


Figure 3.4: Plot of equation 3.1

Bringing it all together, comparing δ to the two random generated numbers results in three possibilities: do nothing, increase the range, or decrease the range. The first comparison checks if the first random number is larger than δ. However, it is expected that in synchronised neighbourhoods the value being compared to is large. It is therefore unlikely that it will increase its range as a result of this comparison. Conversely, the second check performs a smaller-than comparison. The decision making model choosing this path is more likely to be the one that leads to decreases in range.

Breakdown or the detection of change

The fractions of robots going left or right serve as the input to the breakdown detector. The breakdown detector evaluates the rates of change compared to the previous simulation frame in order to detect if the local neighbourhood is less synchronised than before. If it is indeed less synchronised, a flag is raised. Whether or not the flag is raised, along with the fraction variables, it serves as an input to the final stage of the ControlStep. Its implementation is based on the work of Rausch et al [12].

Decision making model

Reaching the end of the ControlStep, a robot must now make a decision based on the results of the preceding steps. The decision making model affects the communication range of the robot and has three results: increase the communication range, decrease it, or do nothing. The fraction variables are reduced to one parameter by taking the difference between them. Pseudo-code 3

offers a way this can be implemented.

```
Input: DegreeLocal, FractionDifference, BreakdownDetected
Result: The robot makes a decision whether or not it changes its range
if DegreeLocal > 0 then
    Random ←RNG(0.0, 1.0);
    if Random>FractionDifference then
        increase range by range step;
    end
    Random ←RNG(0.0, 1.0);
    if Random<FractionDifference then
        decrease range by range step;
    end
end
else if BreakdownDetected then
    increase range by range step;
end
```

Listing 3: Robot decision making model

Spontaneous switching

In line of the research of locust dynamics of Huepe et al [15], noise is introduced where each robot has the possibility to spontaneously switch the direction in which it is moving. This is simply done by comparing a configurable probability with a random generated number. A spontaneous switch merely changes the sign of the velocity parameter of a robot.

3.2.3 Loopfunction

The Loopfunction is less relevant to the emergence of the scale-free network than the Controller. However, it does become important when trying to sample data on a global scale. An extension to a Loopfunction is defined by its PreStep and PostStep functions which are called respectively before and after the ControlSteps of the robots in the virtual world. Subsection 3.2.1 introduced the separation between a local and global degree. The Loopfunction offers an ideal location to keep track of the global part of this problem.

The global graph G

The Loopfunction initializes and keeps track of a global graph G during each simulation frame. Whenever a communication link is determined between two robots, the link is cast into an undirected link and added to the graph[3]. This graph needs to be regenerated at every time-step as certain links may disappear.

The PreStep handles the initialization of the graph and updates the ranges of the robots if the ranges were changed by the decision model. The graph G is then populated by observing the sensor readings of each robot in the world. G now contains every undirected connection and is the global state of the network at that time. The PostStep calculates the degree distribution of the network at that time-step, sorts it in ascending order, and adds it to an array of distributions.

Once the simulation ends, the Loopfunction is allowed to handle cleanup of data. The array of degree distributions is averaged over time. Along with the degree distribution of the final time-step, they are sanitised by only allowing certain values to be recorded and output to distribution files. The data is sorted (either in ascending or descending order) and all values that are equal to zero are omitted from the output. There are two reasons for this decision. First, presenting the data in a sorted way allows the experimenter to more easily see what the minimum and maximum values of the distributions are. Secondly, zeroed values can create problems for the algorithm of section 3.3 as an x_{min} of zero causes the power law distribution to diverge. More importantly, nodes with zero degree are not part of the network and are therefore ignored.

3.3 Fitting a power law to produced data

As the data set is generated through some other means than a generative model such as the BA model, the output of the Loopfunction once the experiment has finished requires further analysis. In order to examine whether a scale-free network emerged, a script is used to check if a power law distribution can be fitted to the distributions in the output files. This section discusses how the empirical observations should be analysed through listing 4 as proposed in the work of Clauset et al [3].

1. Estimate values for α and x_{min}
2. Perform goodness-of-fit test of data in relation to power law based on estimated values, then either accept or reject the hypothesis
3. Compare with alternate hypotheses via likelihood-ratio test

[3]The current implementation uses an adjacency list. As the average degree of the network approaches the number of nodes, an adjacency matrix may be better

Listing 4: Recipe for fitting a power law to empirical data [3]

As mentioned before in section 2.3, if the degree distribution of the network follows a power law, then the network is scale-free. However, there is a caveat to this statement. Proving that a distribution follows a power law can never be concluded with utmost certainty. By using the listing, values for α and x_{min} are estimated and therefore can still be subject to statistical fluctuations. The second step of the recipe tests how reliable the hypothesis is [3].

The tools used to perform power law analysis were a combination of the works of Alstott et al [16] and Nepusz [17] who based their programmatic implementations on the same recipe as discussed in this section. Appendix A contains the script used to evaluate the output file.

3.3.1 Estimating core power law variables

An estimation of the exponent α requires a prior estimation for x_{min}. Assume that the latter value is known, then the value of α can be estimated through maximum likelihood estimators (MLEs). Let n be the amount of entries in the data set and x_i entry i of that data set, then for a data set of continuous (real) numbers the MLE for α can be determined through equation 3.2. There exist separate functions for data sets of discrete values, but they are rather complex. A simpler alternative is to change the continuous MLE slightly to support discrete values. In this case, the integer values are considered real numbers that have been rounded to the nearest integer. This results in the discrete MLE in equation 3.3. The error on both MLEs can be calculated with equation 3.4 [3].

$$\hat{\alpha} = 1 + n \left[\sum_{i=1}^{n} \ln \frac{x_i}{x_{min}} \right]^{-1} \tag{3.2}$$

$$\hat{\alpha} \simeq 1 + n \left[\sum_{i=1}^{n} \ln \frac{x_i}{x_{min} - \frac{1}{2}} \right]^{-1} \tag{3.3}$$

$$\sigma = \frac{\hat{\alpha} - 1}{\sqrt{n}} + O(1/n) \tag{3.4}$$

The MLEs are guaranteed to become more unbiased as the data set grows, especially as it approaches infinity. However, most data sets are finite in size, which is definitely the case in this experiment. As a rule of thumb, estimates are considered unreliable if the sample size is smaller than fifty entries. Bias remains present but decays as $O(n^{-1})$ [3].

Suppose now that x_{min} is not known, then it needs to be estimated as well. Its value defines the start of the power law tail in the log-log plot. Any values of $x < x_{min}$ are not included in the fitting procedure. An accurate estimator of x_{min} is required, as the equations show that the MLEs for α are dependent on it. Figure 3.5 shows the effect of poorly estimating x_{min} and as a result, the erroneous estimates for α. A naive solution to determining x_{min} would be to eyeball the location where the line on the log-log plot starts, but this approach is prone to error.

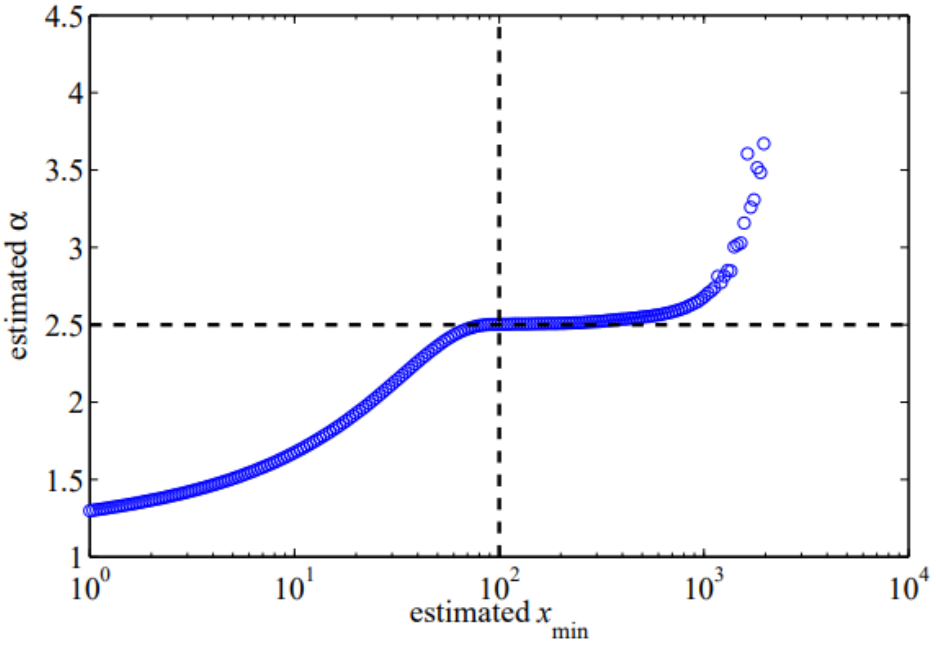


Figure 3.5: Erroneously estimating x_{min} directly affects the estimate of α. Actual values of the power law: x_{min}=100,α=2.5 [3]

A more sophisticated method was proposed by Clauset et al [18] in their work based on the frequency of terrorist attacks which yields accurate estimates for both continuous and discrete data sets. Kolmogorov-Smirnov (KS) tests are performed on the cumulative distribution functions of the empirical observations ($S(x)$) and a potential fit ($P(x)$) which yields the maximum distance between the two. The estimate of x_{min} is chosen for which the result of equation 3.5 is minimised. The result is called the KS statistic. The KS approach is reliable for smaller data sets of at least 1000 entries [3].

$$D = \max_{x \geq x_{min}} |S(x) - P(x)| \tag{3.5}$$

This thesis focuses on the initial understanding of the emergence of power law distributed degrees. As in such an initial state, the analysis must be run many times to gain valuable insights,

it is therefore more useful to start with a smaller sized swarm. However, following the above discussion, in succession to the study presented in this thesis, it will be worthwhile scaling the swarm up to sizes of $N > 1000$.

3.3.2 Goodness-of-fit test

In theory, the method described in subsection 3.3.1 for estimating the core power law parameters can be applied to any data set. Furthermore, this dsupport or rejects the hypothesis that the data distribution follows a power law with the given parameters. Due to the power law hypothesis being applicable to any data set, it can be considered rash or controversial to immediately claim that the fitted data set follows a power law. Purely by statistical fluctuations, a power law distribution may emerge from some system that does not intend to generate one. Deducing that the distribution follows a power law based on the manifestation of a semi-linear shape appearing on the logarithmic scales is not enough. Clauset et al [3] define this as follows:

> "These distributions . . . being roughly straight on a log-log plot is a necessary but not sufficient condition for power law behaviour."

Figure 3.6 aids this statement as it shows a log-log plot of three distributions which visually resembles a linear shape. After observing this plot, this could lead to someone potentially erroneously claiming that all three are power law distributions. However, only the blue plot actually follows a power law. There is a need for a way to quantify whether or not the proposed fit is valid, determining whether or not the power law hypothesis is plausible. The remainder of this subsection discusses a method of quantifying this plausibility.

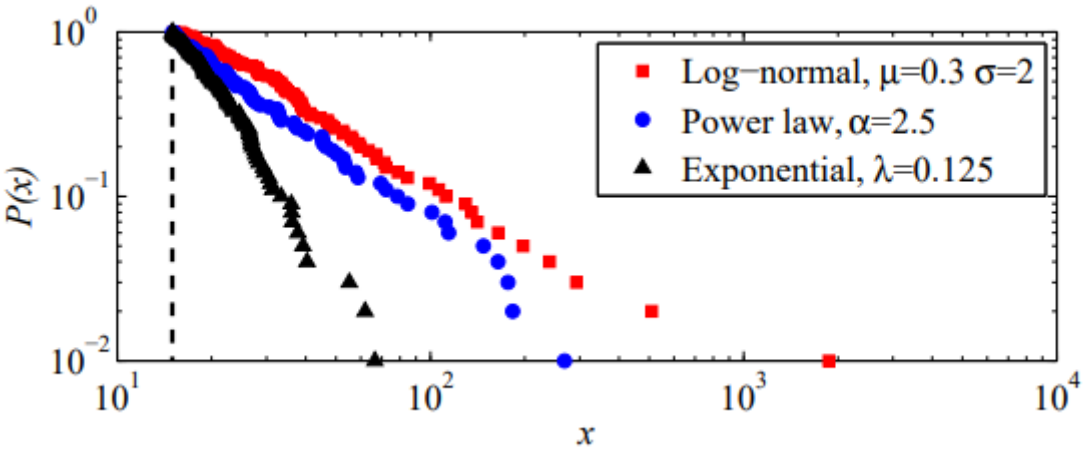


Figure 3.6: The cumulative distribution functions (CDF) of three data sets on the log-log scale [3]

The method as proposed by Clauset et al [3] is the following: first, generate numerous synthetic data sets from a power law distribution function based on the parameters generated from the power law hypothesis. Second, fit the synthetic data sets in the same way as proposed in subsection 3.3.1. Finally, compare the KS statistics of the synthetic sets with the one from the hypothesis. The further away the empirical result is from the synthetic ones, the less likely that the hypothesis of the original data set is valid. Clauset et al [3] point out that the effectiveness of the proposed method is largely dependent on the used method to quantify the distance between data sets. As previously mentioned, some unrelated statistical process may generate a power law distribution by accident, although the probability of this occurring decreases as the sample size increases. In the context of the experiment as proposed by this thesis, the sample size of the degree distribution of the robots is equivalent to the swarm size.

The proposed method is called a goodness-of-fit test which generates a p-value. The p-value quantifies the plausibility of the hypothesis. It is defined as the fraction of synthetic KS statistics larger than the empirical distance. Let s_n equal the amount of synthetic data sets, K the set of synthetic KS statistics derived from their respective distribution, and E the KS statistic of the empirical data set. Then their relation to the p-value can be summarised through equation 3.6. As p approaches one, the disparity between the empirical data set and its true power law model can only be attributed to statistical deviations. Conversely, as p approaches zero, it can be confidently concluded that the hypothesis should be rejected. Note that the synthetic data sets are based on the estimates of the empirical one, but their fitting is independent. The KS statistic is relative to the best fit, rather than the original data set. This ensures that the p-value will be an unbiased estimate [3].

$$p = \frac{1}{s_n} \sum_{i=0}^{s_n - 1} [K_i > E] \tag{3.6}$$

Two questions in relation to the synthetic data sets remain to be answered: how are they generated and how many are required? Clauset et al [3] propose a method for generating a synthetic data set which combines a similar empirical distribution for points below x_{min} and power law following numbers for its tail. Suppose that the empirical data set contains n entries of which n_{tail} entries follow a power law. Then, generate a random number from the true power law model with estimated α and x_{min} with probability n_{tail}/n. Conversely, randomly choose a number x_i from the empirical data set from the region $x < x_{min}$. Repeat this process until the amount of entries in the synthetic data set equals the empirical one. This approach yields a synthetic collection with a power law tail and a similar, non-power law distribution for the empirical observations below x_{min}.

Using this approach, Clauset et al [3] determine that in order to generate a p-value that is accurate within ϵ of the true underlying value, at least $\frac{1}{4}\epsilon^{-2}$ data sets need to be created. As

the requested accuracy for the p-value becomes larger, the value for ϵ decreases, and the amount of required data sets increases exponentially. A software implementation needs to consider the limitations of the p-value's accuracy, or more specifically, the time-space trade-off of requiring additional wasted accuracy. The work of Nepusz [17] uses the same value for ϵ of 0.01 as originally proposed by Clauset et al [3] which, consequently, is also used in this experiment to verify the existence of a scale-free network. An overview of the cost of a more accurate p-value can be seen in table 3.1.

Table 3.1: The price of a higher p-value accuracy. Evaluation of $\frac{1}{4}\epsilon^{-2}$

ϵ	0.1	0.01	0.001	0.0001
required data sets	25	2500	250000	25000000

After generating an accurate estimate for the p-value, a decision must be made. It was previously explained that the p-value offers some insight the more it approaches either zero or one. In most cases, the experiment defines a certain p-value threshold between zero and one. A hypothesis with a p-value under this threshold is rejected, otherwise it is considered sufficiently plausible. Clauset et al [3] set this threshold value at 0.1, which is also used in the experiment as laid out in this thesis, but in some cases this may drop even to as low as 0.05. Depending on the experiment, and the intuition of the experimenter, this threshold value may take any value necessary between zero and one. Applying the goodness-of-fit test on the distributions of figure 3.6 yields figure 3.7. As the sample size increases, the p-values of the red and black distributions start to fall off and it becomes progressively safe to reject their power law hypotheses. The 'true' power law distribution more or less retains its p-value regardless of sample size.

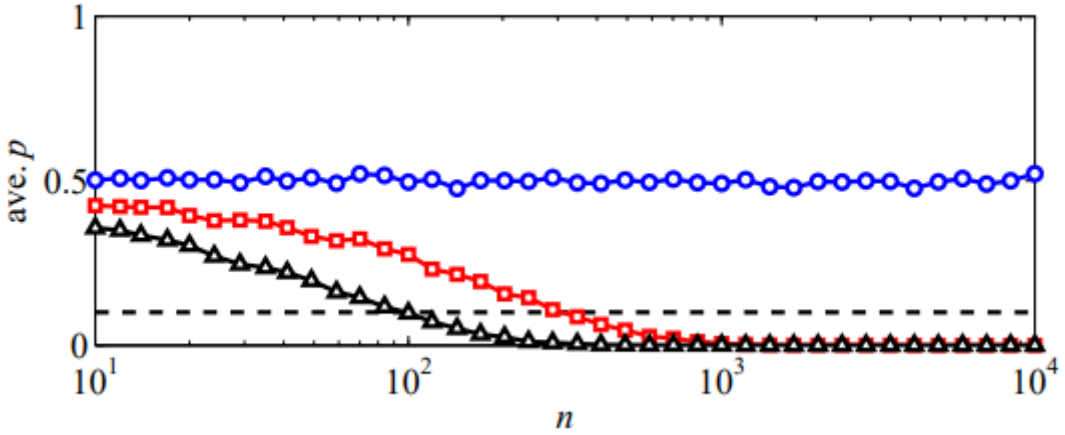


Figure 3.7: The calculated average p-value for the different distributions of figure 3.6. Note that as the sample size increases, the p-value becomes a more reliable metric to rule out a power law hypothesis. A p-value threshold of 0.1 is used [3]

Even after the hypothesis is accepted and the p-value is sufficiently large, there may still be uncertainty that the data set follows a power law. For instance, it is possible that some other distribution is still a better fit than the one suggested by the hypothesis. Subsection 3.3.3 discusses this problem further and offers a method to rule out other competing distributions. Furthermore, Clauset et al [3] mention that as the data set is small, the possibility that the empirical observation follow a power law happens more frequently. This leads to an erroneously large p-value as it tends to approach its upper bound. As mentioned at the end of subsection 3.3.1, in the context of the experiment it may be worthwhile to scale the swarm size to higher numbers. A discussion about this problem can be found in section 4.7.

3.3.3 Likelihood-ratio test

A naive way of ruling out whether another distribution is a better fit for a data set can be achieved by calculating a fit for every possible distribution. The caveat of this approach is that in theory, there are an infinite amount of possible distributions. Thus, a choice must be made. In the case of power law distributions, it may be a good idea to try fitting exponential or log-normal distributions as they all visually resemble each other. Normal distributions need not be ruled out either. Fitting to these other distributions yields a new p-value per fit. Suppose that the fit with the highest p-value to be the best one and eliminate other fits with lesser p-values. Naturally, this should only be done if the original p-value of the goodness-of-fit test of the original power law hypothesis is plausible [3].

A more straightforward approach rather than repeatedly performing numerous Kolmogorov-Smirnov tests between distributions is to perform the likelihood ratio test as proposed by Clauset et al [3]. In a nutshell, the idea is to take the logarithm of the ratio of likelihoods of distributions, then observe the sign, and finally make a conclusion. The sign of this test can be positive or negative or even zero, favoring either distribution, or a tie between distributions, which in itself is unlikely. Taking the likelihood of a distribution is equivalent to taking the product of all probabilities in the distribution:

$$L = \prod_{i=1}^{n} p(x_i) \tag{3.7}$$

Then the ratio between likelihoods is calculated by dividing the likelihoods of two different distributions:

$$R = \frac{L_1}{L_2} = \prod_{i=1}^{n} \frac{p_1(x_i)}{p_2(x_i)} \tag{3.8}$$

Finally, taking the logarithm of the ratio yields the log likelihood ratio ρ:

$$\rho = \ln R = \sum_{i=1}^{n} [\ln p_1(x_i) - \ln p_2(x_i)] \tag{3.9}$$

The sign of ρ is not enough to safely determine which distribution offers a better fit. Considering that it is expected that the expected value of ρ is zero across many data sets, then due to statistical fluctuations, ρ's sign may unexpectedly switch and becomes unreliable. Hence, ρ needs to be adequately far from its expected value in order for the sign to become reliable again. In order to quantitatively determine that ρ is sufficiently far away, it is necessary to know its standard deviation σ [3].

The work of Vuong [19] offers a method of finding the standard deviation of ρ and at the same time generating another p-value. Note that this p-value is not equivalent to the result of the goodness-of-fit tests. If this p-value is lower than a certain threshold (in most cases 0.1), then the method suggests that the sign of ρ is reliable enough to determine which fit is better suited for the data set. Otherwise, ρ's sign is not decisive of either fit. This approach not only gives an additional support for the plausibility of the fit of a power law distribution, but it may also indicate that the data set is not suitable for the power law fit [3].

The work of Alstott et al [16] offers an implementation which uses Vuong's method. In the context of the experiment as laid out by the thesis, this implementation is frequently used to determine if a scale-free network emerges. An overview of the scripts used to fit a data set is given in Appendix 5.

4

Results

Chapter 3 defines the experiment and sets the stage for this chapter. All robots start with the same conditions and parameters, except for their spawn location in the simulated world. At each time step, the robots in the homogeneous swarm have important parameters included in their individual decision-making process. The number of detected neighbours and whether or not breakdown is detected are among these parameters.

First, this chapter will discuss initial observations based on the various runs of the simulation. Second, this chapter defines various robot roles that emerge across simulations and attempts to explain them. Third, this chapter discusses the way the scale-free network is observed. Fourth, a quantitative analysis of the decision making model is presented. Furthermore, methods of quantifying isolation and the subsequent quantitative analysis are discussed. Next, the effect of different swarm sizes is discussed in relation to the effectiveness of the experiment to generate a scale-free network. Finally, the experiment is approached from a theoretical perspective and an effort is made to link it to the generative theoretical models of scale-free networks.

4.1 Initial observations

This section briefly discusses some initial visual observations from various simulation runs. Some of these may not be quantitative but they helped find concrete insights into the way the robots behave and to define these roles. The following section builds upon these observations.

An important early observation was that the robots' ability to change their ranges dynamically, creates the potential to generate a global scale-free network. Conversely, running simulations where setting the range step to zero for each robot removed the emergence from happening, rendering the decision making model moot. It became apparent that the model affecting the ranges of their communication is one of the important pillars of emergence and requires additional inspection.

The world of the experiment resembles a two-dimensional torus in which robots are geographically distributed. Figure 4.1 shows the virtual world of the experiment. As discussed in chapter 3, due to the deterministic nature of ARGoS every run of the same seed will place the robots on the same locations at the start of a run. Another observation follows from the initial distribution. Some robots spawned in a remote location with no discernible neighbours nearby. On the other hand, some robots spawned in regions that were densely populated building visible clusters.

Spatially isolated nodes remained mostly isolated for the duration of a run. However, they did seem to eventually detect several neighbours on the global scale. Either they had a large detection range, or they were detected by remote neighbours with a large detection range themselves. It seemed that these nodes had a large chance of becoming a hub. At the same time, these robots seemed to change their colour at nearly every time step; they repeatedly switched between going left and right. This induced the detection of breakdown as the fractions kept changing between time steps and in turn, caused an increase in the robot's ranges.

The last observation already hints at the emergence of different roles in the simulated environment. The next section aims to define and explain these roles.

4.2 Heterogeneity in a homogeneous swarm

This section discusses the observations in more depth and shows how the decision making model induces a form of heterogeneous behaviour in the swarm. The subsections will define the roles and discuss the product of this heterogeneity more in depth.

Even though the swarm consists of homogeneous robots, some seem to fit into different kinds

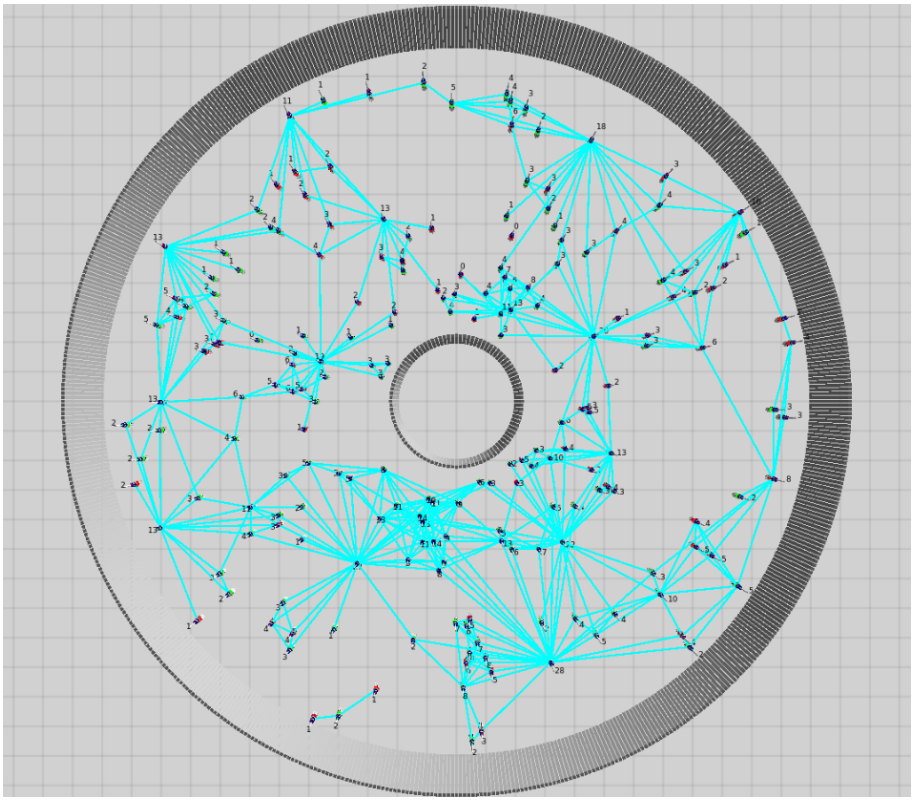


Figure 4.1: A screenshot of the virtual world of the experiment. Each black dot represents a foot-bot. The blue lines between the foot-bots represent the (directed) communication links

of roles which leads to different forms of behaviour. One of the major driving forces behind this behaviour stems from the distribution of locations of these robots at the start of a simulation run. As the robot locations in relation to one another affect the decision making of the respective robots, it in turn directly affects the range. Therefore, the change of ranges leads to the possibility of the robots connecting with one another, affecting the degrees of these robots, which may ultimately result in a scale-free network at the global-scale. As such, there is an intimate relationship between the range and degree distributions.

First, in order to concretely define the roles, various hypotheses were established from the empirical observations discussed in section 4.1. Additionally, parameters such as the range step, lower bound, etc. were adjusted. Finally, in order to aid the refinement of these hypotheses and reach concrete roles, additional code was written to allow the user to mark certain robots as relevant or interesting by index[1]. Robots that are marked interesting would have information about their internal events output to a separate file, which was in turn analysed after numerous runs. The recorded events are related to the robot's decision making model as described in subsection 3.2.2. An example of such an interest file can be seen in listing 5.

```
Starting interest output for node 192 seed 12
2: Decreasing range due to RNG < fraction_difference(1)
2: Spontaneous switch of direction
3: Decreasing range due to RNG < fraction_difference(1)
7: Increasing range due to RNG > fraction_difference(0.333333)
```

Listing 5: An example of the output in an interest file. Note the different time steps at the start of each line

The following subsections build upon the initial observations from the section 4.1.

4.2.1 Isolated hubs

An isolated hub is a node which is spatially isolated from the rest of the nodes at the start of the simulation run; it initially spawns in a region of the simulated environment without any discernible close neighbours. It starts off with a degree of zero but manages to gain the status of a hub after several time steps of the simulation. After becoming a hub, it and its neighbours visually resemble a star network with the hub at the center. These hub also tended to end up with a high communication range. Multiple isolated hubs may emerge but they are limited in number due to their very nature[2].

[1] ARGoS will assign the same ID per robot per seed

[2] Alternatively, numerous hubs could exist but the network will no longer be scale-free

Initial assumptions and discovery

From initial observations, it looked as if these nodes rapidly change their direction at almost every time step due to changing color. The assumption was made that this could be related to the spontaneous switch probability causing the node to change its direction. Because the detection of change only takes notice of the direction of the robot, and fractions of neighbours going left or right, it induces a detection of breakdown and leads to the increase of the robot's range. Therefore, the hypothesis was made that these isolated nodes caused their range to increase due to measuring themselves switching, erroneously thinking that there might be a breakdown.

However, these nodes' intensity of spontaneous switching was significantly higher than expected from random behaviour. The probability for spontaneous switching was set to 0.1, whereas the nodes switched at every time step. This discovery lead to writing code which allows one to define a number of nodes that output their events to a file. Numerous interest files for isolated nodes showed that the range increase was indeed due to detecting breakdown and being isolated, but not as a result of a spontaneous switch.

Another interesting observation was what happened once the robot did in fact detect neighbours: it still claimed it has no neighbours and it would resume with its unique behaviour of increasing its range almost indefinitely. Initially, it was believed that this was a bug of ARGoS itself, but as the following will show it is in fact expected behaviour.

Explanation of behaviour

In order to give an explanation of this behaviour, the decision making model introduced in chapter 3 was evaluated step by step. At the start of the simulation, the robot did not detect breakdown and it decides to make a decision. It checks its sensor readings and detects zero neighbours causing the AverageVelocity calculation to be dependent on only one variable: the random generated number that is added to this average velocity. However, before this variable is added, it is evaluated that AverageVelocity will always be zero for robots that detect no neighbours. This results in an average velocity of 0.5 meaning that the direction of an isolated robot defaults to the right and thus FractionRight will be filled in with 1.0. As long as the robot keeps defaulting to the right, the breakdown detector code will not fire and the robot keeps its range and remains isolated. But the question still remained, what is the effect of the random number?

The randomly generated number falls in between -1.0 and 1.0 inclusive which is then added to the average velocity. In order for the robot to suddenly detect a change of direction, the RNG

needs to generate a number lower than -0.5. This occurs nearly 25% of the time for each time step in the case of a node that is isolated and causes the robot to move to the left. The changing of direction causes the breakdown detector to detect a surge of change, which in turn leaves the isolated node no other option but to enter the BreakDownDetected part of the DMM, which finally causes the robot to increase its range. The problematic part is that such a robot defaults to the right at every time step, creating a flip-flop between going left and right. This positive feedback loop repeated itself until the robot detected a neighbour locally.

As mentioned in chapter 3, ARGoS draws directed links between a robot that knows of the existence of another robot. The confusion stemmed from the experimenter visually not being able to discern between an undirected and a directed connection. On a global scale, directed links are replaced by undirected links in graph G. As such, these isolated nodes do not in fact detect neighbours but get detected consistently by remote neighbours.

Increasing the range step may result in these nodes getting detected sooner, but may cause the issue of every node being able to communicate with every node (on a global scale). As the network becomes more and more connected, some networks can lose its intermediate scale-free status as the degree of all nodes starts to reach the swarm size sooner.

There is a caveat, once an isolated node happens to physically move close enough into the neighbourhood of one or multiple nodes, they will detect a non-zero local degree which affects the decision making model as well as the breakdown detector in a completely different way. In a sense, a parallel can be drawn to hubs in general: getting too close to an isolated hub can lead to a form of targeted attack. The change of the local neighbourhood may induce changes in its range and in turn cause a hub to lose a lot of links in a short time frame. The effects on a global scale may be impactful. This change of behaviour may lead to the roles as described in the following subsections.

4.2.2 Initial hubs in synchronised clusters

If an isolated hub can be considered as one side of a coin, then robots that have a high degree from the start would be the other. Some hubs may emerge from the first time step as they spawn in densely populated regions. Therefore, they immediately detect a lot of neighbours in their close vicinity. ARGoS effectively decides that there is a doubly directed connection between this robot and its neighbours. But as time progresses, they lose their hub status, and they are unable to compete with the intensity of the range increases of an isolated hub. Moreover, their range, degree and neighbours stay relatively unchanged during the simulation run. Due to their high degree count at the start of a simulation, they get the name of an initial hub.

From initial observations, the nodes with high degrees initially spawned in densely populated

regions. Clustering of these neighbourhoods is easily seen by the naked eye. Clusters visually resemble and have a high chance of being a connected component of the global network. Due to many overlapping ranges, most if not all nodes in this region are physically close enough to communicate with each other. Thus, ARGoS adds links in both directions. The degrees in this local neighbourhood remain close to the local average for the duration of the simulation. The communication ranges of nodes in this neighbourhood tend to be small on average.

Most of the nodes in the region around the initial hub also tended to move alongside it. Their colour did not change as sporadically as isolated hubs meaning their movement is also not as unusual. This lead to the assumption that the local neighbourhood is synchronised. This is related to the discussion about the calculations of fractions and δ in subsection 3.2.2.

4.2.3 Hubs in asynchronous clusters

Between the previous roles lies the following more subtle role. Certain nodes spawned in clusters but did not behave as expected. They emerged as initial hubs but were able to compete with the high degrees and ranges that isolated hubs exhibit by aggressively increasing their range as well; their ranges and degree counts were considerably higher than the cluster they are situated in. However, even though their global degree increased, they were only able to detect physical neighbours on a local scale. What separates this role from the initial hubs mentioned at the start of the section is the state of synchronisation of the local neighbourhood. Asynchronous hubs emerge in asynchronous clusters.

As discussed above, the value of δ in relation to the randomly generated numbers heavily determined the robot's decision. The inverse happened: δ approached its minimal value of zero which causes an increase in the likelihood of the decision model choosing to increase the robot's range. The more asynchronous the neighbourhood becomes, the more the fractions become similar. As long as the neighbourhood remained asynchronous, it would either choose to increase its range or do nothing at all. It was possible that the neighbourhood became synchronised again when most nodes moved in the opposite direction out of the detection range of this robot.

4.2.4 Final remark

A final remark to make is that these roles are not set in stone for the entire duration of a simulation run. For example, isolated hubs had the ability to enter a densely populated region which would heavily affect their decision making. The hubs would evaluate their physically close neighbours in order to detect breakdown or to detect whether the local neighbourhood was synchronised or not. At that moment, the isolated hub became a hub in a cluster which completely changed the way it behaved. These small changes at a local scale potentially affect

the global scale-free nature of the network, as the loss of many hubs could indicate that the scale-free network has disappeared.

4.3 Observing the scale-free distribution

Following the introduction of the various forms of data generated by the experiment at the end of subsection 3.2.3 and using the statistical methods of section 3.3 to try and determine if a data set follows a power law, this section briefly applies these methods to some of that data.

The degree distributions across numerous seeds can be output in two different ways. First, the experiment will always output the state of the network at the end of the simulation. The simulation ends at a certain discrete time step t_{end} when the duration of the experiment exceeds that time step (defined through a configuration variable), the experiment unexpectedly shuts down (more than likely due to a crash), or ends at the request of the experimenter. Consider this version of the output at time step t_{end} to be a snapshot of the state of the network equivalent to the data mentioned at the end of subsection 3.2.3. An alternative version of the degree distributions is calculated during the entire run of the simulation. At every discrete time step, the snapshot of the degree distribution at that time is sorted ascendingly (or descendingly) and added element wise to a collection of real numbers. Reaching t_{end} then averages the degree distribution out over that same time variable. From this point onward, κ and λ distributions are used to denote snapshot and over time distributions.

The reason for a distinction between the κ and λ distributions is threefold. The κ distribution between t_{end} and t_{end+1} can differ significantly due to the dynamic nature of the experiment. Observing a power law distribution in this data does not guarantee that the same can be said of its previous or next time step. It nevertheless can still offer valuable insights. As the λ distribution is calculated by sorting the κ distribution at each time step, the coupling to the robots responsible for that entry in the data set disappears. Therefore, the distribution becomes a global quantity of the experiment and loses information on the local scale. Furthermore, taking the average over the run of the experiment reduces statistical fluctuations and offers a more reliable result. Finally, the probability of a node having a time averaged degree of zero decreases the larger t_{end} becomes. Non-zero elements are valid when dealing with fitting to a distribution, thus there is no longer need to omit any entry in the data set. In short, observing a power law pattern in λ distributions is a more reliable metric than in its counterpart.

Taking the output across numerous seeds yields the same amount of entries per distribution. In order to be able to verify the results, the same seeds of one to thirty are used. By adding all distributions together and taking the average across the number of seeds reduces the statistical fluctuations even further. Taking a look at the log-log plot of this distribution can already hint

at scale-free behaviour. Figure 4.2 shows multiple examples of the log-log plot of the λ and seed averaged degree distributions for different swarm sizes. As always, observing the presence of a semi-linear plot on the logarithmic scales is a necessary but insufficient indicator of a power law distribution.

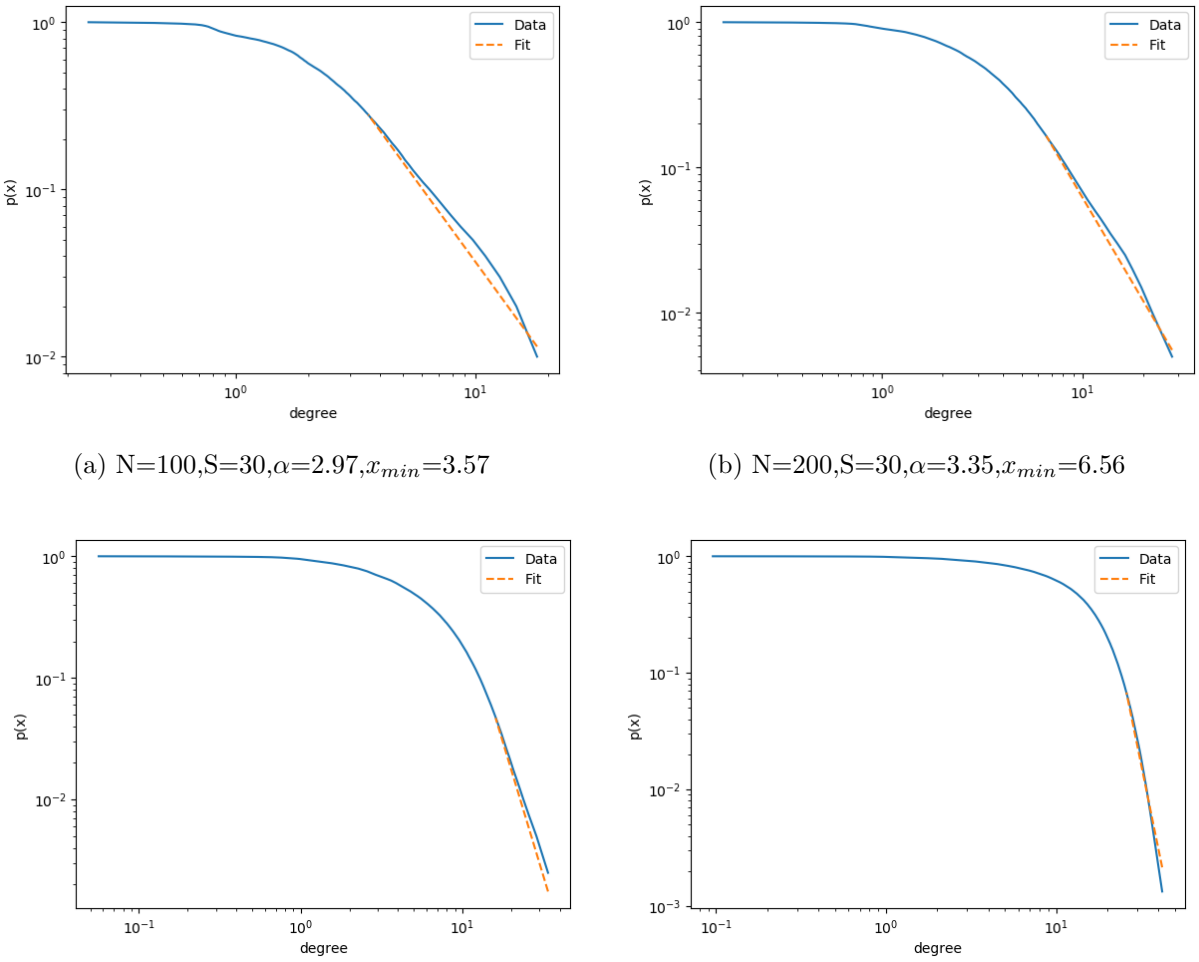
Generating the λ and seed averaged distributions permits the use of the methods described at the end of section 3.3. From this point forward, the power law evaluator is equivalent to the implementation of those methods in the form of a script such as the one given in appendix 5. As the name implies, the power law evaluator evaluates if a given distribution follows a power law. As mentioned before, the script used to perform the fitting and goodness-of-fit test utilize the same p-value threshold of 0.1 to accept the power law hypothesis. The same script also makes use of Vuong's p-value threshold of 0.1 in order to conclude if the sign of the log likelihood test can offer a final answer (and whether both distributions are suitable). The result of the fitting shows that the semi-linear log-log plot was indeed indicative of a power law tail. Therefore, a scale-free network emerges from the experiment and can be observed in the power law tail of κ. λ can be scale-free but it does not necessarily imply that there was a scale-free network at any point in time as merely due to statistical fluctuations, it can happen that the distribution follows a power law.

A note of caution concerning the power law evaluator in general. As the experiment deals with a relatively small number of nodes, the distributions will consequently also contain a relatively small number of entries. The statistical methods are better defined for larger data sets. Further testing is required by scaling the experiment to even larger swarm sizes.

4.4 Analysis of the Decision Making Model

Section 4.2 alludes that the decision making model is largely responsible for the emergence of various behavioural roles. This section will analyse this component to determine which parts of it are required to keep the scale-free nature of the degree distributions and conclude whether or not the DMM itself is a necessary part of the experiment.

During its ControlStep, the robot will act as a result of the decision made by the DMM. The pseudo-code 3 of subsection 3.2.2 shows that the DMM can in essence be reduced to a ternary decision path: potentially respond to changes of the local neighbourhood based on readings received from detected neighbours, respond to the detection of breakdown likely as a result of the robot switching directions or of a dynamic neighbourhood, or decide to do nothing. It would be more appropriate to reduce it further to a binary decision, omitting the third path. In order to quantify the necessary components of the DMM, the following strategy in listing 6 is proposed:



(a) N=100,S=30,α=2.97,x_{min}=3.57

(b) N=200,S=30,α=3.35,x_{min}=6.56

(c) N=400,S=30,α=5.40,x_{min}=15.81

(d) N=750,S=30,α=8.13,x_{min}=25.96

Figure 4.2: The λ and seed averaged degree distributions of the experiment for increasing swarm size N but constant seed count S visually show a semi-line appearing on the logarithmic scales (blue line). Fitting this data set with the power law evaluator results in the orange dashed line. Note that although the estimates deduced by fitting become more and more extreme as the swarm size increases, with power law exponents far exceeding the theoretical limits of the BA model, all of these distributions pass the power law evaluator

1. Introduce configuration variables in code which allow enabling or disabling of the local neighbourhood and breakdown checks in the decision making model
2. The swarm has a homogeneous DMM; all robots share the same configuration variables
3. Run the experiment across 30 seeds and output relevant data, alternating between enabling and disabling one or both components of the DMM
4. Evaluate the distributions using the power law evaluator
5. Deduce probability of a scale-free network emerging as a direct result of the DMM averaged over the amount of seeds

Listing 6: DMM analysis strategy

The default case of the DMM up until this point was to be fully enabled for both components. Therefore, the analysis is completed by running the experiment for the other three cases. Disabling both decision paths of the DMM effectively lobotomizes a part of the robots controller. Range adjustments no longer occur and will remain at the default value as specified by the configuration variable loaded at the start of the simulation. It is expected that the only network that can possibly emerge from this state is a random geometric network, consequently with an upper bound on its global radius of one (depending on the configuration value).

It is believed that the initially isolated nodes have the tendency to increase their range the most out of all of the different roles or at the very least are predisposed as breakdown occurs at roughly 25% for each of them. Therefore, in the case of the DMM which has its breakdown detector enabled, the experiment should still see isolated hubs emerging as their ranges rapidly increase. Conversely, merely allowing local neighbourhood checks may also lead to the rise of hubs. As mentioned in section 4.2, local environments that behave asynchronously may yield a hub as well, although the odds of this occurring will probably be less than the other case. Taking a glance at figure 4.3 visually confirms these assumptions for at least one seed as it shows the effect of different parts of the DMM.

Table 4.1 provides further support of the visual observations. By evaluating whether the distribution follows a power law over various seeds, it can be observed that the fully equipped DMM reliably generates a scale-free network. Omitting the local neighbourhood check is not as detrimental as removing the breakdown detector. Only allowing the local neighbourhood checker to exist shows that the experiment tends to favor the case in the DMM where the ranges decrease as a result of synchronised local neighbourhoods. The breakdown detector seems to be a sufficient component for generating scale-free networks in the context of this experiment, but the combination of both will yield optimal results.

As section 4.2 discusses, the breakdown detection of a controller is a major contributor to the

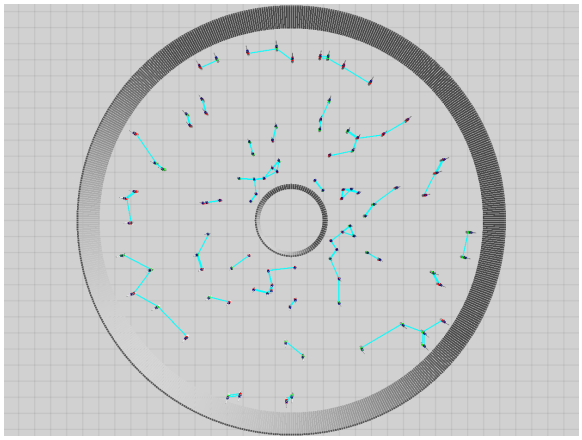
emergence of the isolated hub role. Without it, these nodes have no concrete means of increasing their range until they get swallowed up by a cluster or simply another neighbour getting close enough to physically detect them. The analysis presented in this section should strengthen the weight of the DMM in aiding the experiment generate a scale-free network and shows that the combination of both components yields the most stable results.

4.5 Quantifying isolation

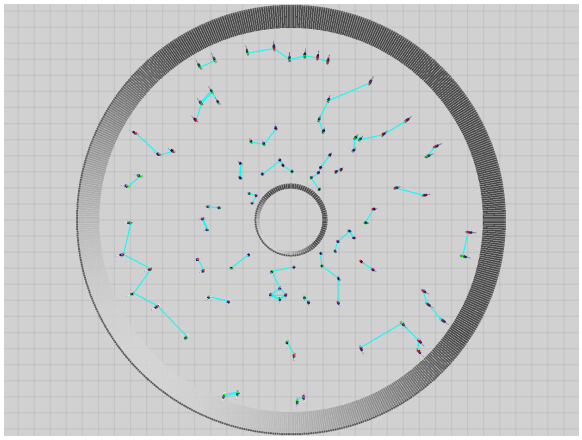
Section 4.2 accentuates that isolated nodes and clustered nodes form important behavioural roles in the experiment. The question remains: how to objectively quantify isolation and non-isolation? Isolation is related to a node's Euclidean distance[3] to its nearest neighbour. The higher this metric is in relation to similar measurements for the remaining robots in the experiment, the higher the probability that that node can be marked as isolated. On the other hand, the lower this metric is, the better of an indicator that the robot is in a clustered environment.

In order to aid the experiment answer this question, the simulator is expanded with finding nearest neighbours during the PostStep method of the Loopfunction. In essence, for each robot the tentative closest distance to all possible neighbours in the world is calculated and saved. Rather than calculating the actual Euclidean metric, the square magnitude is kept in memory, as taking the square root of a number is an expensive operation and performing it numerous times can be considered wasteful in the time critical simulation step. Moreover, a comparison between two square magnitudes is equivalent to a comparison between their respective square root values. The evaluator is implemented naively with $O(n^2)$ behaviour and a method of implementing it is presented in listing 7. Furthermore, code was written to visualize all nearest

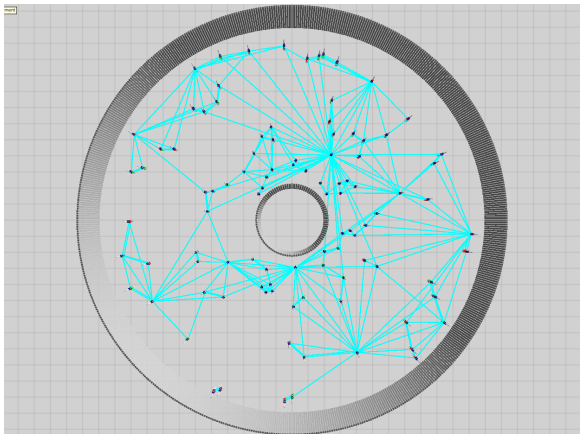
[3]Defined as the distance between two points or robot R and another robot O in world space: $\sqrt{(x_r - x_o)^2 + (y_r - y_o)^2}$. The z component is shared among all robots and ignored



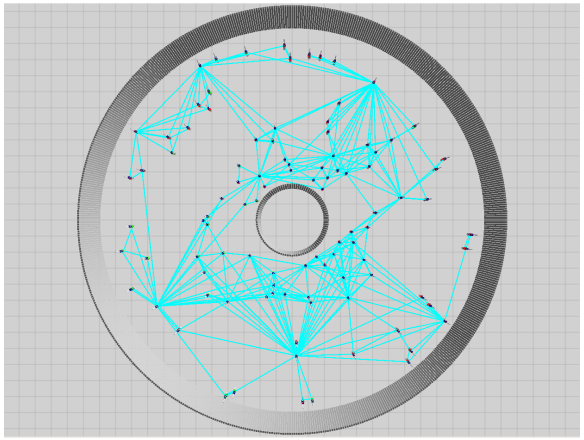
(a) DMM disabled



(b) DMM with local neighbourhood check



(c) DMM with breakdown check



(d) DMM enabled with both checks

Figure 4.3: The visual effect of enabling or disabling certain parts of the decision making model (N=100, seed 10, $t_{end} = t_{100}$) a) Experiment running without a DMM per robot. There are some robots that manage to be connected to one another, but the maximum distance between two connected robots will be the initial range b) Experiment running with the local neighbourhood check enabled for the DMM. Behaves similarly to not having a DMM at all c) Experiment running with the breakdown check enabled for the DMM. Visually, a more complex network seems to emerge along with some nodes that are isolated, yet are highly connected d) Experiment running with the DMM fully equipped. Simlar to c), a complex network emerges with isolated nodes. Clustered regions are more nuanced and may have hubs as well

neighbours in the world to aid the research even further as visualised in figure 4.4.

```
Input: Robots
Result: Each robot knows of its nearest neighbour and the square magnitude distance to
        it
foreach Robot ∈Robots do
    Robot.NearestNeighbour ←NULL;
    smallestDistance ←MaxReal();
    nearestWorldLocation ←Vector3(0,0,0);
    foreach OtherRobot ∈Robots do
        if Robot ≠OtherRobot then
            distance ←EuclidianDistance(Robot.WorldPosition, OtherRobot.WorldPosition);
            if distance < smallestDistance then
                smallestDistance ←distance ;
                nearestWorldPosition ←OtherRobot.WorldPosition;
            end
        end
    end
    Robot.NearestNeighbour ← smallestDistance, nearestWorldPosition;
end
```

Listing 7: Naive nearest neighbour evaluation

Although the naive implementation of listing 7 is adequate enough to not cause an extensive bottleneck during simulation as the swarm sizes remain relatively low, its usefulness will decrease when the swarm size is scaled up to larger numbers. Other more efficient methods to determine nearest neighbours, such as spatial localisation[4], may offer faster results. Alternatively, the nearest neighbour evaluation can only be done after longer time intervals, as the robots do not move intensely between time steps and the nearest neighbour of an isolated node will more than often remain identical. However, for clustered environments this can rapidly change.

By adding another line of code to the drawing section of the simulation, it becomes possible to display the actual range of a robot as a red circle around it as can be seen in the various sub-figures of figure 4.5. Once again, the figure highlights the complex evolution of the experiment and further shows that the κ distribution of the degrees at t_i is potentially subject to various changes. The figure also demonstrates that isolated nodes tend to increase their range quite aggressively (the three nodes in the Western quadrant), nodes in clustered regions tend to decrease their range, and a hub with large range can also emerge in a clustered region (node

[4]However, this method might not be sufficient when dealing with isolated nodes in simulations with a relatively small swarm size, as the distance between this node and the remainder of the robots exceeds the boundaries and capabilities of other implementations. Further discussion is outside of the scope of this thesis

Table 4.1: Probabilities of scale-free networks emerging in relation to enabling or disabling parts of the decision making model. Enabled and disabled indicate the status of both components (N=200, averaged over 30 seeds)

	Disabled	Local neighbourhood check	Breakdown check	Enabled
Probability of SFN (κ)	0.2	0.067	0.2	0.67
Probability of SFN (λ)	0.0	0.0	0.7	1.0
Average degree	1.78	0.32	5.63	6.89
Average range	1.0	0.5	1.41	1.43

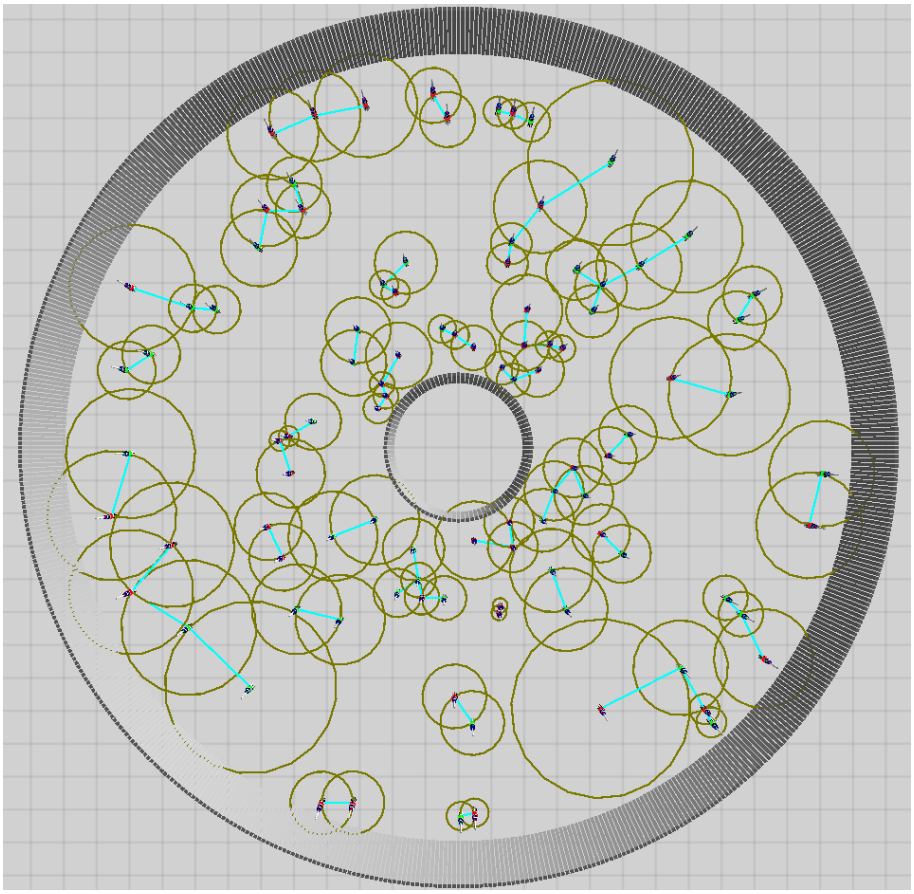
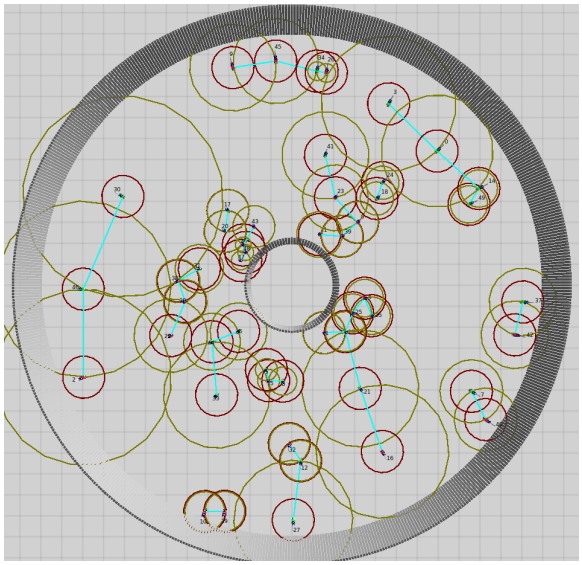
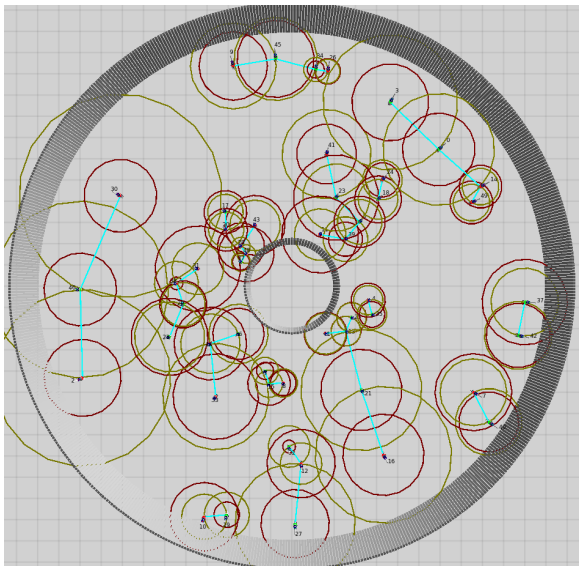


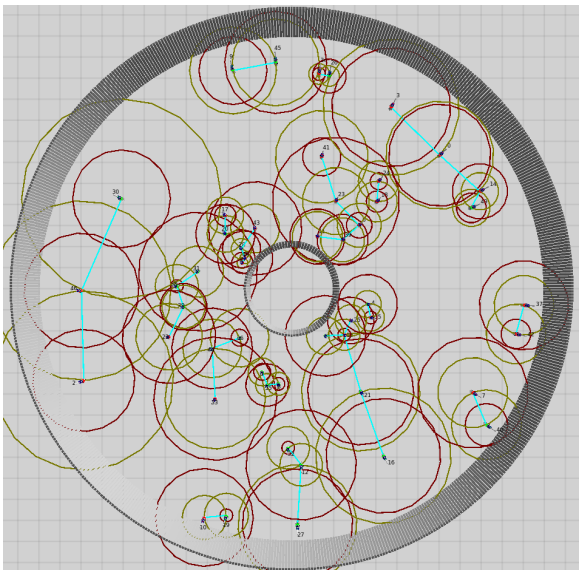
Figure 4.4: Each robot has a certain distance to its nearest neighbour. Around each robot a circle is drawn, where its radius indicates the nearest neighbour metric. Other robots only exist on the borders of that circle, but never closer than that outer edge. Note that the blue links in this picture specifically signify the relation between a robot and its nearest neighbour, which consequently is bounded by the circle. Note how the larger and smaller circles allow for an objective definition of isolated and clustered nodes (N=100,t_0)



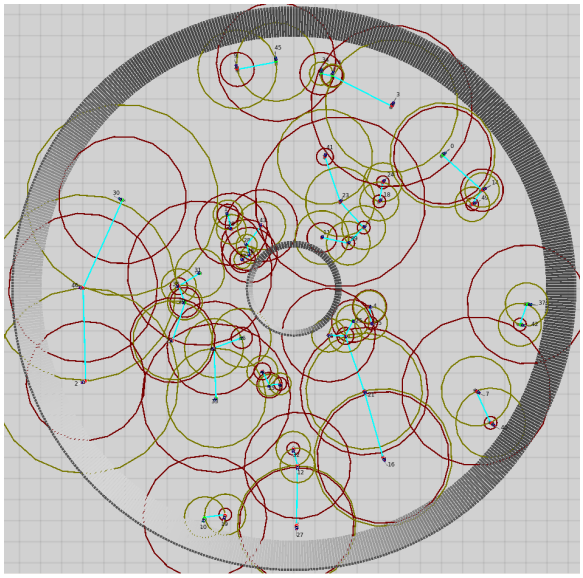
(a) Experiment at t_0



(b) Experiment at t_{10}



(c) Experiment at t_{20}



(d) Experiment at t_{30}

Figure 4.5: The evolution of the network during its first time steps. The green-yellow circles indicate the nearest neighbourhood void, while the red circles are representative of the robot's communication range. Note that the initial ranges are homogeneous but rapidly change as a result of the decision making model's choice for each individual robot

with ID 23 in the North-Western quadrant). Section 4.6 continues the research by using the newly introduced nearest neighbour distance as a heuristic to find representatives of the roles of section 4.2.

4.6 Evaluating the roles

This section will try to quantitatively evaluate the roles as presented by section 4.2. It becomes necessary to find representatives of these roles in the population and analyse them further by taking a look at the various pieces of data that they produce during the run of the experiment. The produced data is related to metrics already introduced in the previous sections. During the run of the simulation, each robot keeps track of its history per time step. Each time step, a history entry is generated with its local and global degrees, whether it detected breakdown or not, the distance to its nearest neighbour, etc. and added to its history container. In order to find the representatives and analyse their behaviour in relation to the observed roles, the following strategy is proposed:

1. Decide on a heuristic to select nodes from the population in order to be representatives
2. Predict what the found nodes should represent, its behaviour, development of its data, etc.
3. Observe the data associated with the found nodes. Either confirm observations, find new insights, or find exceptions to the proposed roles

Listing 8: Role representative strategy

A heuristic is a method that can be used to facilitate finding a solution to a problem. In this case, the heuristic is used to quickly find nodes from the population that may be representatives of one of the three roles introduced. The question remains: what should they entail? The following subsections present two possible methods.

4.6.1 Nearest neighbour distance heuristic

The nearest neighbour distance heuristic (NNDH) is based on the findings of section 4.5. In essence, the distance of a node to its nearest neighbour can help define its isolation or clustering. The initial isolation was described to be an important factor in the emergence of an isolated hub. Similarly, the two other roles are defined by their clustered nature. Selecting three nodes by this metric from the start of the experiment may result in finding a representative. The heuristic sorts the population by the nearest neighbour distance descendingly, then selects the

nodes at both extremities and the robot at the center of the sorted collection. From this point onward, the n_{min} and n_{max} relate to the extremities, while n_{mid} relates to the center node.

It is expected that an emerging isolated hub is isolated at the start of the experiment. It should be able to reliably increase its range due to detecting breakdown numerous times. n_{max} as selected by the NNDH and should behave in a similar way. n_{min} spawned in a more densely populated neighbourhood. As the swarm size increases, the probability of multiple nodes spawning in the initial range of n_{min} rises as well, i.e. in a clustered environment. Depending on the neighbourhood during the simulation, it can either emerge as an asynchronous hub or as an initial hub but failing to retain its hub status. It is more difficult to predict for n_{mid} what its behaviour should be, thus it becomes necessary to analyse its data. n_{mid} may be relatively isolated or, in another seed, be relatively clustered.

4.6.2 Post-degree heuristic

Alternatively, the post-degree heuristic (PDH) uses a different metric in order to choose representatives. Furthermore, selection is executed by operating on the knowledge of the simulation after it has run. Consider the global degree of a node. A hub is defined as a node with an exceptionally higher degree than the average in the network. Conversely, there are many other nodes of which the degree is significantly lower than that of the hub. Observing the behaviour of these nodes may result in new insights or even find a representative. By running the simulation once, due to the deterministic nature of ARGoS, the end result will remain the same for the same set of parameters. Therefore, it becomes possible to cache the IDs of the robots with the highest and lowest degree counts at the end of the simulation and select them. Similarly to the previous heuristic, the robot in the center of the sorted collection is selected.

In the case of the PDH, the n_{max} will contain the highest global degree of the network at the end of the simulation. Therefore, it is guaranteed to be a hub and requires a strong increase of its global degree during the simulation. It is also expected that it contains a higher range than its initial state or a relatively high nearest neighbour distance. n_{min} is guaranteed not to be a hub, it will more than likely be a node with a small range in a synchronous environment and as such, a small nearest neighbour distance. n_{mid} is more than likely not a hub as the nature of scale-free networks dictates that hubs are sparse.

4.6.3 Analysis

The comparison between n_{max}, n_{mid}, and n_{min} is done as these samples differ strongly in terms of the two metrics used by the heuristics. The history data of the selected nodes by the heuristics form data sets that can be analysed. The progression of this history data can be found in figures

4.6, 4.7, and 4.8. Each vertical collection of graphs is bound to the same node. Omission of data only happens in the breakdown graphs of each node at the bottom of each page. A missing breakdown entry indicates that breakdown was not detected, a breakdown of 0 indicates that breakdown was detected, and a breakdown of 1 indicates that breakdown was detected and lead to increasing the robots range. A missing value for δ indicates that it was not calculated at that time step, whereas an existing value of δ should be approached with the discussion of figure 3.4. The direction decision metric is equivalent to either 0 or 1, which was deduced from the sign of the robot's velocity. This subsection will discuss each vertical collection separately.

n_{min} selected via NNDH starts in a clustered environment. In the initial time steps, its own range increases slightly as a result of an slight asynchronous environment. After this initial increase, the range falls off rapidly to the lower bound of 0.3 signaling that the environment has become more or less synchronised. There are still however some fluctuations in the range. The degree reaches its lowest point around t_{40} after which it begins to rapidly increase its range as a result of another asynchronous local neighbourhood. The simulation ends at t_{100} but the graphs indicate that this node will more than likely truly reach hub status after this point.

The n_{mid} of the NNDH spawns in a less clustered environment. Its nearest neighbour has a very strong range increase, however, the node itself decreases its range to the lower bound. The readings of δ indicate that at some point another synchronised cluster is currently existing at that point in time. At some point, the node loses its connection to its nearest neighbour leading to a singular range increase as a result of detecting breakdown. The local neighbourhood becomes asynchronous again leading to a rapid range increase of the node from t_35 onward. t_60 marks the beginning of another decrease of its communication range. The same can be said about its nearest neighbour.

n_{max} via NNDH is selected due to its initial remote location in relation to other nodes in the experiment. This node achieves hub status at the end of the simulation. As the communication links discussion of section 3.1.2 presented, this node does not detect other robots locally but is detected by many on the global scale. This changes slightly towards the end of the simulation where it does enter the communication range of another robot. Consequently, the behaviour of the robot changes as it now needs to worry about the value of δ. Up until that point, it almost exclusively detects breakdown due to its continued isolation leading to a stark and almost unbounded increase of its communication range. The data set is a strong indicator of the isolated hub role emerging from the experiment.

Moving on to the next heuristic which selects nodes based on the state of a finished experiment. The n_{min} selected through PDH does not have a strong degree count throughout the simulation. It is able to initially detect some neighbours. Its nearest neighbour is showing the strongest signs of increasing its range. As long as it detects this nearest neighbour, this node tends to decrease its range as it is in a synchronised environment. Due to the strong range increase of its nearest

neighbour, it might be the case that these nodes are relatively close to one another, but together are relatively isolated from the rest of the population. The selected node does not become a hub due to it thinking it is in a synchronised environment for the better part of the initial time steps. At some point near t_{50}, both nodes detect another robot with presumably a large range. After this point, even the strong nearest neighbour begins to consistently decrease its range. A strong indication of a synchronised environment absorbing the isolated micro cluster.

The n_{mid} of the PDH spawns in a semi-clustered region and manages to acquire a few links by increasing its range in a moderately asynchronous region. Suddenly, the robot disappears off the grid and becomes isolated leading to rage increases due to detecting breakdown. Before the end of the first half of the experiment, the node enters a synchronised cluster and reduces its range as a result. The second half of the simulation shows a moderately asynchronous region in which the node's range fluctuates.

The final node n_{max} of the PDH is determined to be a hub at the end of the simulation. It shares a lot of its properties with the n_{max} selected by the NNDH. Its closest neighbour is initially close but lacks the range in order for the selected node to be able to do a local neighbourhood check. That nearest neighbour is connected to this node but also to another synchronised cluster on the opposite side of it (relative to the selected node). As soon as δ gets evaluated, the selected node's range begins to decrease as a result of a synchronous neighbourhood.

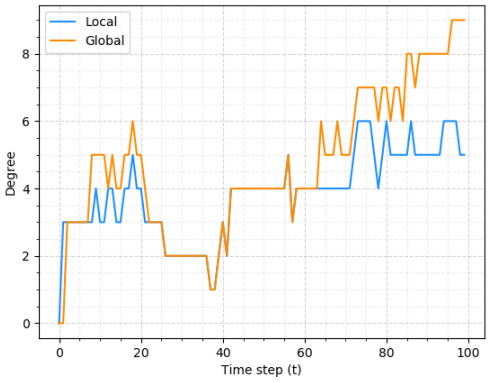
In summary, the observed roles of section 4.2 can be quantified through the use of the history data sets of nodes selected by the two proposed heuristics. The data sets highlight the dynamic nature of the determined roles as well as the fuzzy transition during the defined states. Selecting the max node from both heuristics can help with finding isolated hubs. Of course this analysis is done in the context of a single seed and future work should evaluate history data over many seeds in order to reduce statistical fluctuations. In order to avoid confirmation bias, the heuristics can be augmented by introducing a form of randomness in which the selection procedure separates the population in three parts using a predefined fraction value, and then selects randomly[5] from each respective part the n_{min}, n_{mid}, and n_{max}.

4.7 Density and its effect on the experiment

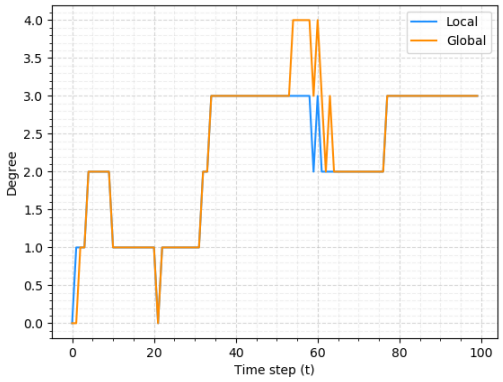
This section discusses the concept of density in relation to the experiment, how changing the swarm size affects various metrics and the probability of the emergence of a scale-free distribution, as well as how this correlates to certain parts of the decision making model.

In the field of swarm intelligence, density is an important concept. There are various forms of

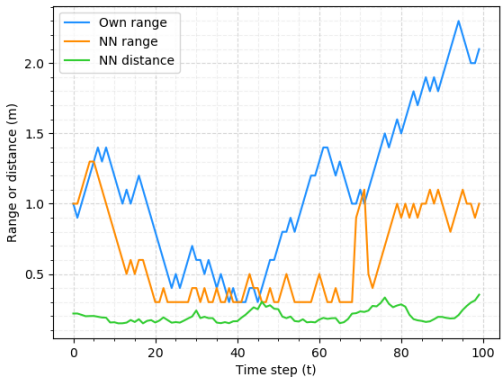
[5]Either unweighted or weighted depending on how close the node is to either extreme or midpoint



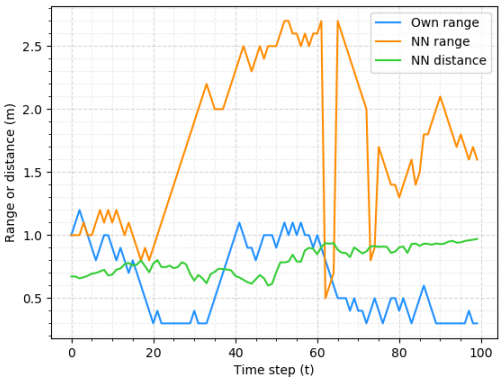
(a) Progression of NNDH min node degree



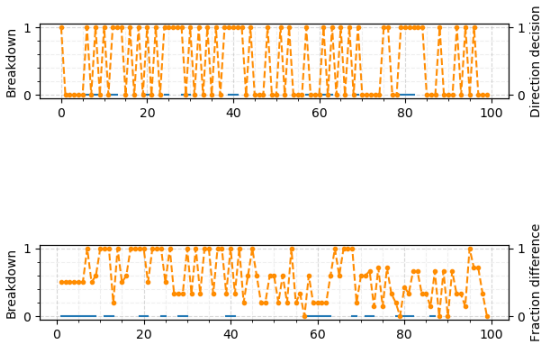
(b) Progression of NNDH min node degree



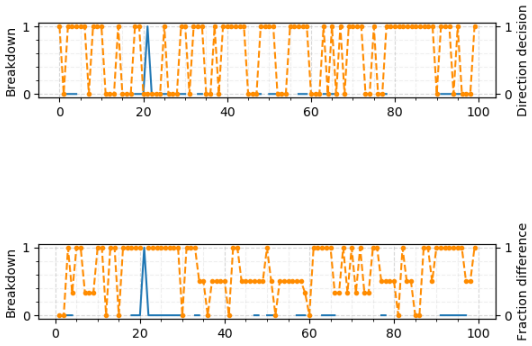
(c) Progression of NNDH min node range distance



(d) Progression of NNDH mid node range distance

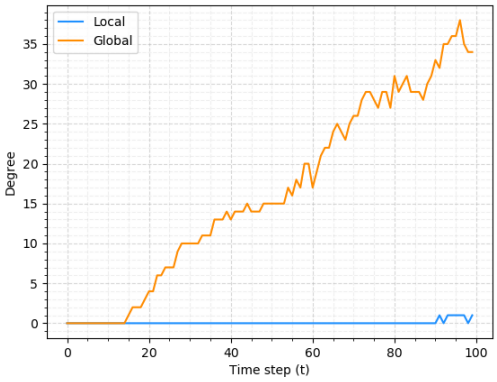


(e) Progression of NNDH min node breakdown

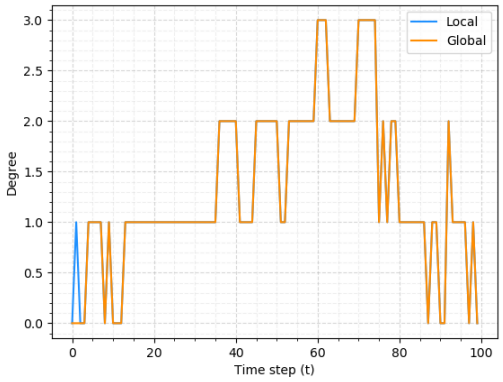


(f) Progression of NNDH mid node breakdown

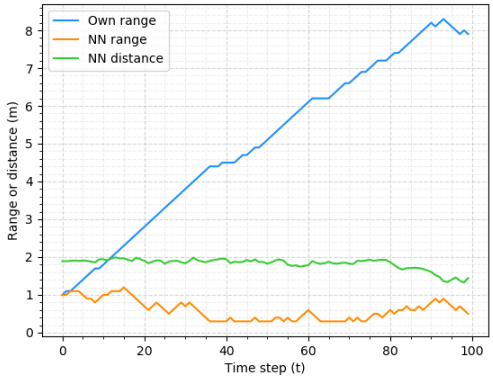
Figure 4.6: The observations of the min and mid nodes via NNDH. For graphs e) and f), the blue line is indicative of breakdown (N=200,seed=10,$t_{end} = t_{100}$)



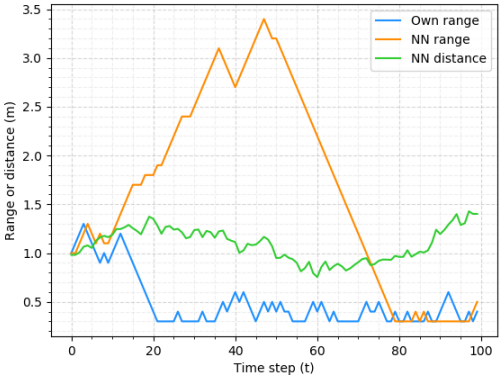
(a) NNDH max node degree progression



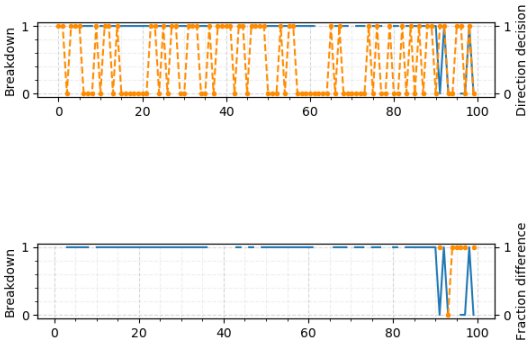
(b) PDH min node degree progression



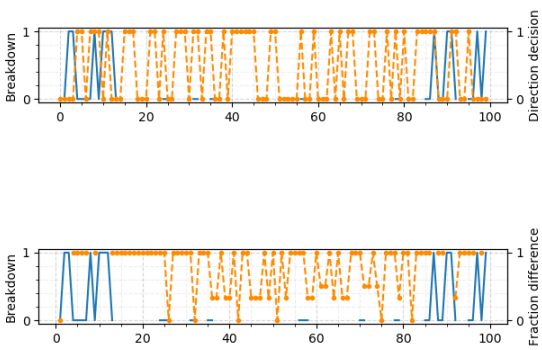
(c) NNDH max node range distance progression



(d) PDH min node range distance progression

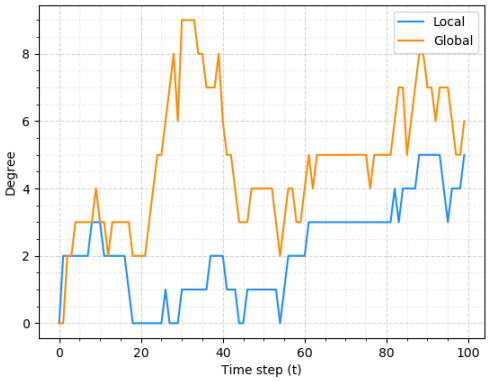


(e) NNDH max node breakdown progression

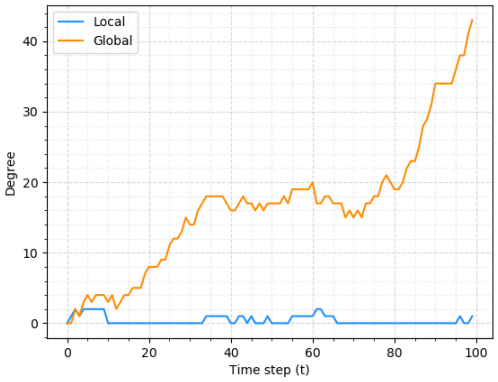


(f) PDH min node breakdown progression

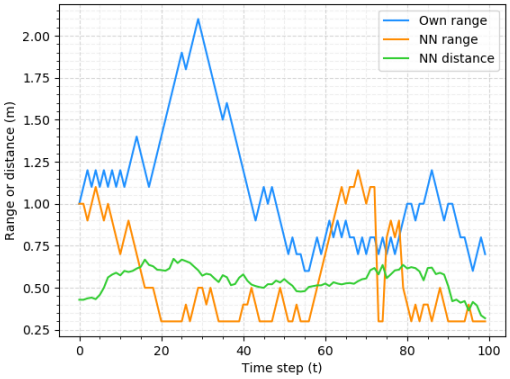
Figure 4.7: The observations of the max node via NNDH and min node via PDH. For graphs e) and f), the blue line is indicative of breakdown (N=200,seed=10,$t_{end} = t_{100}$)



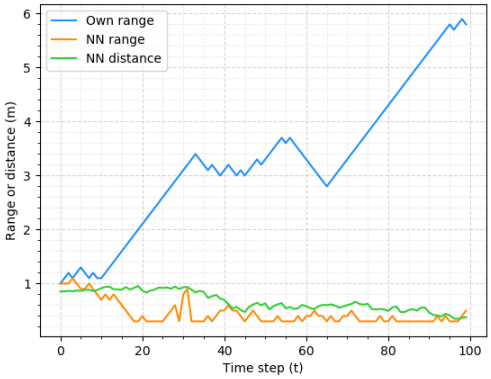
(a) PDH mid node degree progression



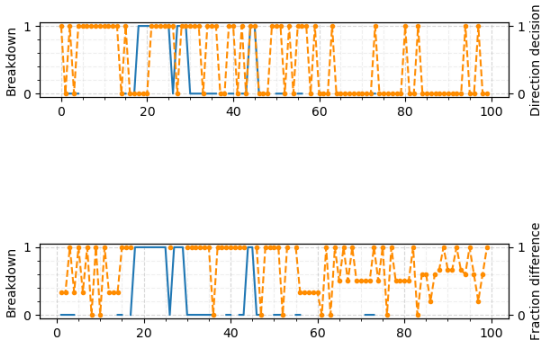
(b) PDH max node degree progression



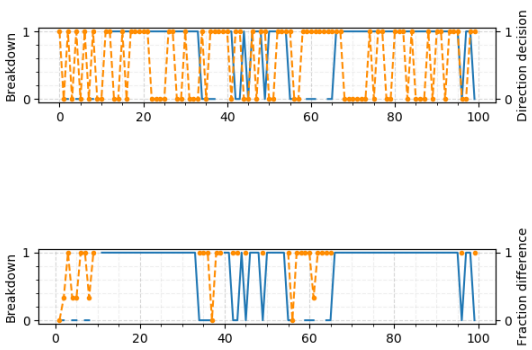
(c) PDH mid node range distance progression



(d) PDH max node range distance progression



(e) PDH max node breakdown progression



(f) PDH max node breakdown progression

Figure 4.8: The observations of the mid and max nodes via PDH. For graphs e) and f), the blue line is indicative of breakdown (N=200,seed=10,$t_{end} = t_{100}$)

density, such as physical or communication density. Physical density can be understood as the amount of agents in the collective system inhabiting a certain section of the world. This form of density has already been frequently used in the case of robots moving in clustered regions. Communication density can be attributed to the amount of communication links existing in the network.

In both cases, density is dependant on the swarm size of the collective system. So far, this thesis has not distinguished its findings based on swarm size alone [6] and assumed a constant population size $N = 200$. It becomes imperative to observe what happens to the various distributions as a direct result of a changing swarm size. However, the swarm size still remains constant during a simulation run.

It is expected that changing the swarm size directly affects certain parts of the decision making model where certain components of the DMM can become moot as previously observed in 4.1. This can then result in the loss of the scale-free degree distributions as observed in section 4.4. The larger the swarm becomes, the more geographically close robots become and as such, fewer nodes remain isolated. The role of the isolated hub will start to matter less and fewer will exist across the simulations. Robots will more easily detect other robots in the vicinity and will therefore only focus on the local neighbourhood check. It was already observed that the decision making model putting focus primarily on the local neighbourhood check performs sub-par. As a result, the breakdown detector becomes less apparent.

On the other hand, a simulation with a sparsely populated environment will see more isolated nodes appearing. Breakdown will be detected more often, leading to robots with larger ranges and possibly larger degree counts. The part which evaluates the local neighbourhood falls away. The decision making model utilising only the breakdown detector still manages to reliably produce a scale-free network. There is an important caveat to this statement: the smaller the swarm size, the less reliable a power law fit becomes. Therefore, fitting should be met with appropriate scepticism for the smallest swarm sizes.

In order to evaluate the effect on the distributions due to dynamic swarm sizes, the following strategy 9 is proposed:

1. Let the swarm size N assume values between 50 and 500 with steps of 50 units
2. For each swarm size, run the experiment across 30 seeds and output relevant data
3. For each seed, run the power law evaluator and cache the result of the binomial test
4. Average the results over 30 seeds to get the probability of the distribution following a power law

[6] Although this was briefly hinted at in section 4.3

Listing 9: Density evaluation strategy

The result of this evaluation can be found in figure 4.9. As can be observed for the λ degree distributions, the smaller the swarm size becomes, the probability of generating a scale-free distribution starts to fall off. The same can be said for increasing swarm sizes. Experiments run with populations between 150 and 200 robots almost guarantee to produce a scale-free degree distribution in the context of the experiment. The ranges of the network form an important part of the experiment and have shown to aid in the emergence of a scale-free network. Likewise, the collection of generated data sets is expanded by outputting the κ and λ range distributions. For small swarm sizes, the λ range distributions do not follow a power law. As the population increases, the range distribution almost guarantees to follow a power law. The zone between 200 and 300 nodes forms an optimal zone in generating λ power law distributions. The λ ranges following a power law is correlated, but is not a necessity to generate a scale-free network in the context of this experiment. This emerging property of the range distribution can be attributed to asynchronous hubs continuing to indefinitely increase their ranges whereas the majority of the network implicitly enforces some upper bound on their ranges.

Inspecting the average values of the experiment over different seeds and population sizes also offers interesting insights. The average degree increases continuously with increasing swarm size. This makes sense as the spatial distribution of the robots becomes denser. Robots initially start with a range of 1.0m and therefore will detect more neighbours at the start of the experiment. The initial degrees of the robots will therefore be higher. Robots that have detected others are more likely to remain connected to them. Conversely, the average nearest neighbour distance and average ranges decrease. The average nearest neighbour metric decreasing can be attributed to the geographic density filling up more with other robots and thus, is a natural evolution of that metric. The average range decreases at a faux-plat rate close to a value of 1.5m.

It has been mentioned before that scaling the experiment to larger swarm sizes (especially when dealing with more than 1000 agents) should see more accurate results of the power law evaluator. Due to the nature of part of the decision making model falling away as a direct result of the spatial density increasing to unhealthy levels, it may be the case that the experiment will guarantee to fail to produce a scale-free network when dealing with such high population sizes. Consequently, the time that it takes to finish a simulation step increases linearly due to ARGoS architecture which may be detrimental to finding results. A proposed solution to this problem may be to increase the arena of the simulation, as the initial spatial distribution of the robots in the world is based on a spherical uniform distributor which randomly places the robots in the world. Currently, the robots and that distributor are bounded by the defined arena of the experiment but it may be necessary to increase those boundaries or to test the experiment by abstracting it even further from locust marching by completely removing the arena.

4.8 Reducing and longrunning the experiment

In this final section of the thesis, the experiment is discussed in a different manner by approaching it from a hypothetical thought process. An attempt is made to link it to the Barabási-Albert model introduced in 2.4. Further data is generated and discussed based on this new approach.

The BA model of section 2.4 offers a theoretical method of generating a scale-free network. It is an abstract approach in which there is no concept of a physical environment, nearest neighbours, or any form of interaction between the nodes beyond deciding which links get preferentially linked to one another. The experiment is more concrete, as it lets the nodes in the network communicate with one another and make decisions together on the local scale. The experiment was not created with the BA model in mind.

In section 4.3, it was previously demonstrated that for low swarm sizes averaged over many seeds, the power law exponent approaches the interval that is well defined for the BA model. Do note that the exponents acquired through fitting through the power law evaluator should be considered with scepticism, as fitting may induce false positives in observing a power law for small distributions. Approaching the experiment as if it were based on the BA model in mind may lead to some new insights.

In a hypothetical situation, suppose that the experiment was built on the BA model as an underlying layer. Furthermore, suppose that it is possible to reduce the concrete model of the experiment to its underlying model by redefining the current implementation in function of the two generative components of the BA model, namely preferential attachment and some form of a growth process. Then it becomes interesting to observe the experiment in a different light.

How can the experiment be reduced? No form of preferential attachment exists in the experiment as intended by the BA model, but robots may already be implicitly preferentially linking as a result of another metric. If the range of a robot is an indicator of how strong it is able to connect to other robots, due to its reach being larger overall, then its degree count should be higher than robots that have a smaller range. Conversely, nodes with a small range will have trouble connecting to their physically close neighbours. What about a growth process? This forms a roadblock as the population size N remains constant during the simulation, but the BA model is defined in function of these two components and requires both of them. The question arises: can the BA model function without a well defined growth process?

The same question was postulated in the work of Albert and Barabási [7] which bases its answer on their previous work in Albert, Jeong and Barabási [20]. In their works, they attempt to generate scale-free networks by omitting a growth process, opting for a constant node size N, and keeping preferential attachment. It was observed that although initially the experiment generates a scale-free network, as the time of the experiment approaches the square of the

population size, $t_{end} \simeq N^2$, all nodes become connected and the scale-free property is lost.

So far in this thesis, all of the research presented stops at $t_{end} = t_{100}$. It becomes compelling to observe what happens to the experiment beyond this threshold value in relation to the observations of Albert, Jeong and Barabási. Do note that their observations were made with the intention of creating a process that generates scale-free degree distributions, whereas the experiment is hypothetically reduced to a model with preferential attachment and without a growth process. In the case of a swarm size of $N = 200$, then $t_{end} = t_{max} = 40000$. Observing what happens in the simulation up to that point may offer new insights. As running the experiment for 40000 time steps takes some time, it is called a longrun.

The experiment already allows the configuration of t_{end} through a configuration variable, so letting the experiment run as long as that variable is trivial. Concerning the files that are already generated at the end of the simulation: is there any risk of overflowing numbers, especially in the case of the λ degree distribution as the κ degree distribution is added at each time step? As the maximum number of degrees is limited by the swarm size N and t_{end}, the upper bound of this relation is $O(Nt_{max}) = O(N^3)$. For $N = 200$ this leads to a maximum degree sum of eight million before averaging over t_{max}, far below any danger of a C++ float overflow error. Therefore, the experiment can be run with no issue. A final note before discussing the results of this simulation: there is no guarantee that the data generated as a result of a longrun will coincide with the observations of Albert, Jeong and Barabási.

The data generated by this experiment can be found in figures 4.12 and 4.13. Both figures show the detected lowest, highest, mean, and median values for their respective metrics recorded at each time step. There are a couple of key observations that can be seen from a first glance. First, the mean and median registered values are highly correlated in both data sets. At almost any point, the mean is slightly larger than the median value, indicating a very soft positive skew for the underlying κ distribution [7].

Second, the highest observed degree at each time step has a stark increase in the early stages of the experiment hinting at the emergence of at least one hub. After this initial stage, the observed highest degree value fluctuates drastically[8]. This behaviour can be attributed to the dynamic nature of the experiment, where robots with a high degree count can suddenly lose a lot of links between various steps as a result of decreasing range, or robots moving out of their bounds. These fluctuations also demonstrates how a κ distribution has the potential to vary immensely between two succeeding time steps.

Third, the mean and median of the recorded degrees also show a stark increase at the start of

[7] Skewness is generally calculated by $\frac{\mu - v}{\sigma}$ with μ the mean, v the median, and σ the standard deviation

[8] Longrunning the experiment over various seeds and then averaging this data should normalise the fluctuations, but takes a considerable amount of time even with the visualisation of ARGoS disabled

Table 4.2: Various minimum, maximum and average values over the t_{40000} records (seed 13)

	Low	Median	High	Mean
Degree minimum	0	0	0	0
Degree maximum	11	32	120	32.64
Degree average	0.87±0.91	11.2±3.73	74.95±18.29	12.91±3.66
Range minimum	0.3	0.4	1.0	0.90
Range maximum	1.0	5.5	200.9	6.56
Range average	0.30±0.01	1.33±0.63	86.82±47.18	2.44±0.73

the simulation. The average value is heavily influenced by the value of the highest recorded degree. Similarly, the lowest recorded value also influences them, and due to a relatively low median in relation to the highest recorded value, half of the recorded values are relatively low as well.

Fourth, the lowest value assumes the value of zero or one most of the time. There are however, two peaks in their data set at t_{9000} and t_{30000}. This directly influences the peaks of the mean and median values. It is peculiar that around those time steps, the mean and median values of the range data sets also exert small peaks indicating another form of correlation. The peaks make sense when considering the intimate relation between ranges and degrees in the network. If the lower bound on the ranges in the network start to increase for all nodes, then more nodes will be able to communicate with other robots in their network, leading to an overall increase in their degrees. Once the peak falls off range-wise, the same applies to the lower degree recordings.

Fifth, the highest observed ranges fluctuate as well, but not as drastically as their degree counterpart. The increases are to be expected with the behavioural roles, especially in the case of isolated or asynchronous hubs. There seems to be a stark rise at certain points, followed by crashes similar to the stock market. The crashing value falls off more slowly than the rapid rising of the largest range. At a point where a peak is reached, the system may be in a state where every robot is in an ideal location to each other and the ranges of all robots dynamically adjust themselves in order not to waste unused range.

In all cases, the initial stage highlights that the original choice for the end of the simulation, $t_{end} = t_{100}$, is adequate enough to show the crucial steps of the scale-free network emerging. Somewhere around this point, the system converges and the various metrics start to oscillate around their respective average values. An overview of these averages can be found in table 4.2.

In order to test the connectedness theory, a depth-first search (DFS) is performed on the global graph G at the end of the simulation in order to determine the components of the graph. It is expected that only one giant component remains that contains all nodes of the graph. After performing the DFS test at the end of the longrun, it was observed that the giant component

indeed contained N nodes. Note again, that this is equivalent to a snapshot and can therefore yield a different result in the t_{end+i} snapshot.

Finally, putting the degree distributions through the power law evaluator should give another indication of what kind of scale-free network exists at t_{end} if any. As previously mentioned, it is more interesting to take a look at the λ degree distribution. Figure 4.11 presents a proposed fit. Although the exponent of the fit is considerably higher than the defined limit of the BA model, interestingly enough the goodness-of-fit test returned a p-value of exactly 1.0.

In summary, the observations of the longrun give a strong indication that the early end of the simulation is valuable in yielding insights into the developmental stages of the scale-free network emerging in the experiment. Even though the κ distribution differ between time steps, the scale-free network emerges as a global property through the λ distribution even after running the experiment for a long time.

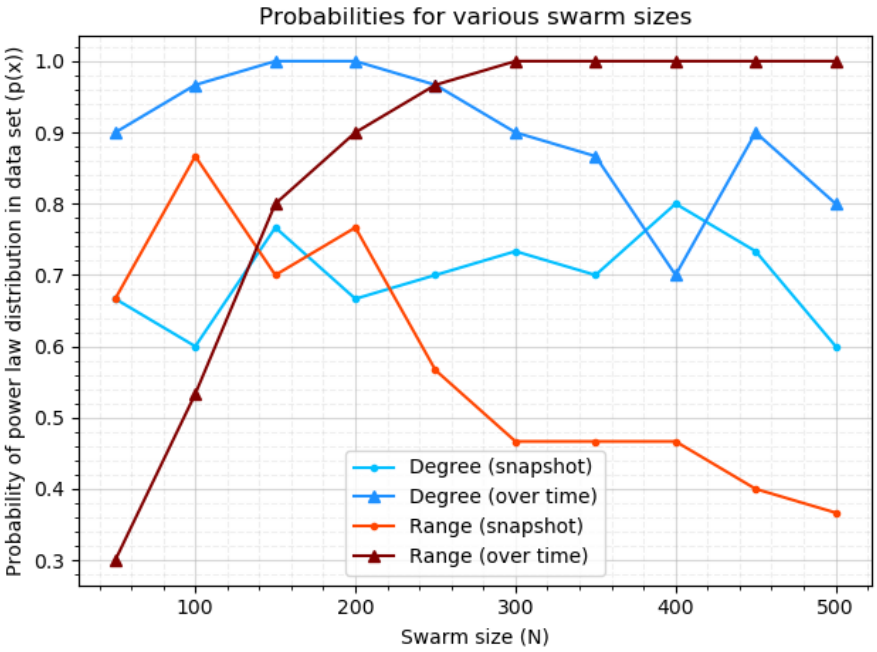


Figure 4.9: Probabilities of distributions following a power law averaged over 30 seeds. The swarm size N is increased by 50 and the distributions generated at the end of the simulation are tested with the power law evaluator

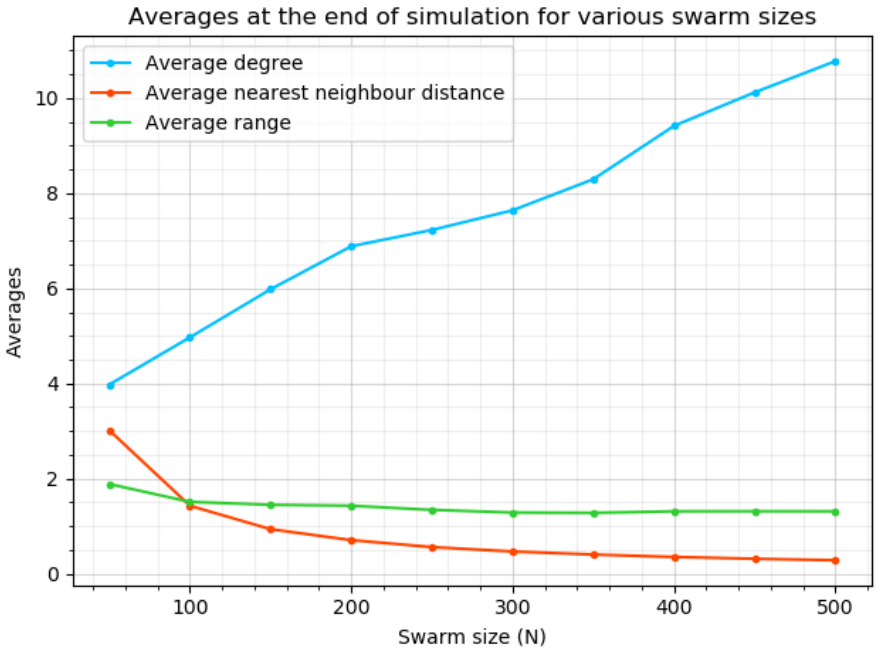


Figure 4.10: Averages of various metrics at the end of the simulation averaged over 30 seeds. The swarm size N is increased by 50.

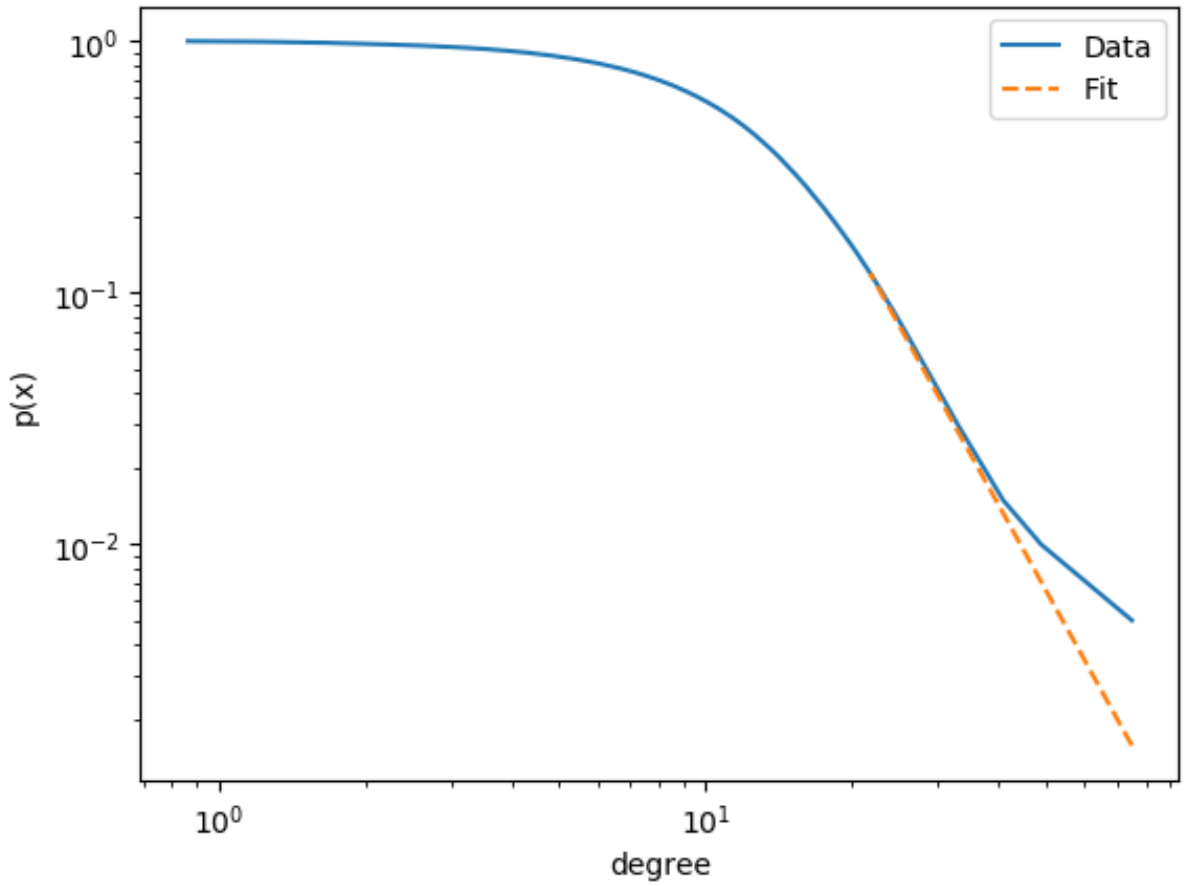


Figure 4.11: Log-log plot of the λ degree distribution of longrun t_{40000} with a proposed fit ($\alpha = 4.49, x_{min} = 21.75$,seed 13)

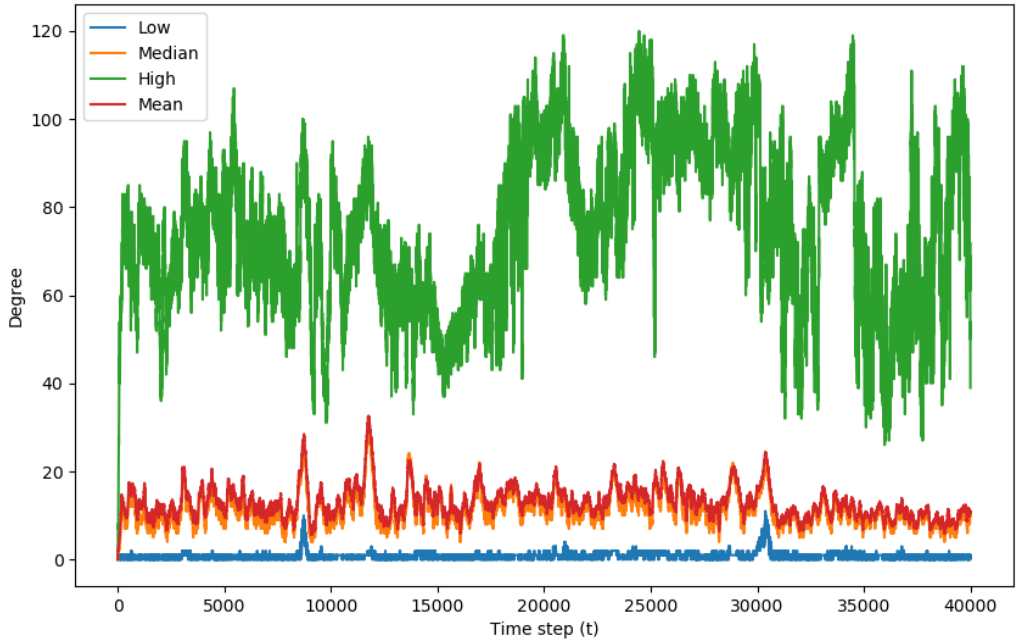


Figure 4.12: Degrees in the network over a longrun of t_{40000} (seed 13)

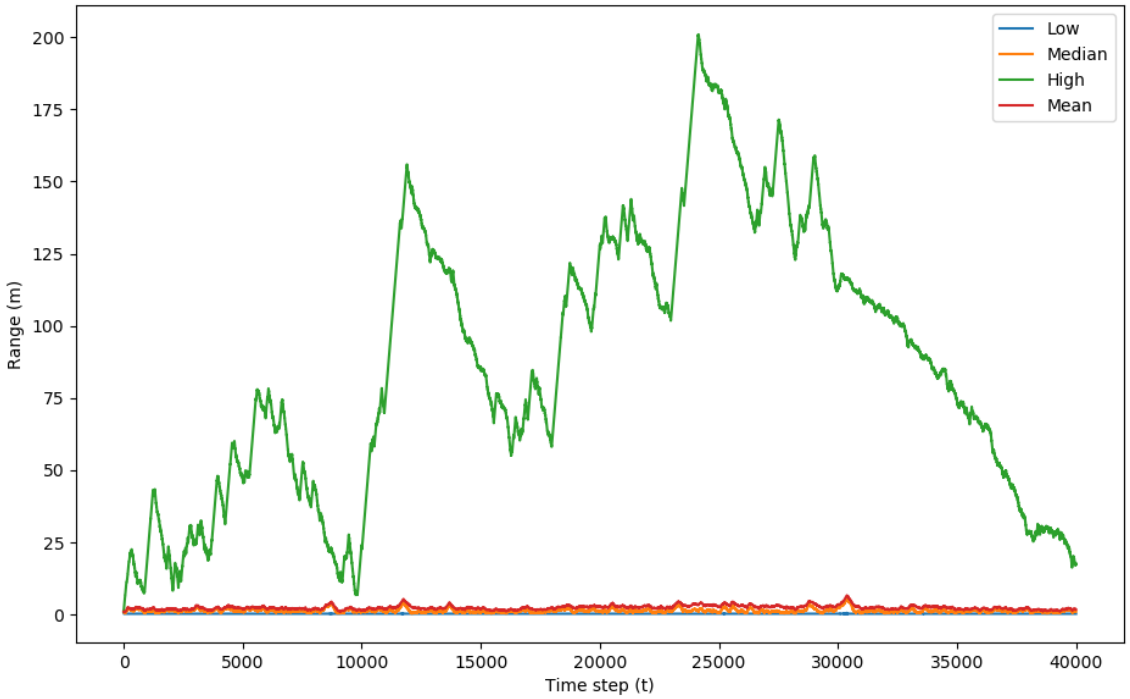


Figure 4.13: Communication ranges in the network over a longrun of t_{40000} (seed 13)

5

Conclusion

The research as presented in this thesis aimed to discover the factors that contribute to the emergence of a scale-free network in the context of an existing experiment in the field of swarm intelligence. It was observed that the degree distribution of the swarm follow a power law on the global scale, whereas the agents of the swarm only interact locally. Based from initial empirical observations, it was deduced that heterogeneous behaviour emerged from a swarm with initial homogeneous conditions.

The various forms of behaviour were then formalised in the form of three roles. These roles were quantified by analysing the history data of nodes selected through heuristics. Furthermore, it was determined that the emerging roles were a direct result of the decision making model that each robot has, each behaving differently based on the state of the local neighbourhood around an agent. This state is dependant on the range of a node, indicating an intimate relationship between the range of an agent and its degree on a global scale.

Analysis of the decision making model showed that it is a required component of the emerging scale-free property. Without it, the experiment does not generate the complex behaviour of a scale-free network. In particular, the breakdown detector is a fundamental contributor to the range and subsequently degree distribution of the network.

Furthermore, the experiment was scaled to various swarm sizes indicating that the reliability of a

scale-free network emerging is also largely dependant on the initial population. In a similar vein, it was also observed that the distribution of communication ranges in the network guarantee to be following a power law the larger the swarm becomes.

As frequently mentioned throughout the thesis, the proposed results need to be approached with some skepticism. Claiming that distributions follow a power law can never be concluded with utmost certainty and still remain difficult to prove. But the resiliency of the experiment to create a global scale-free property emerging across many seeds is not somethings that should be discarded.

Throughout this thesis, it has often been said that the swarm size should assume higher values in order to more accurately test the power law hypothesis. Future work should try to observe what happens when scaling the experiment to vastly higher numbers but should be aware of the limitations and the usefulness of running such an experiment.

This thesis lays heavy focus on the context of the experiment. It would be an interesting endeavour to try to abstract the generation of the scale-free network from the experiment and try to reproduce the same behaviour in another context. It is recommended that some similar form of the decision making model be implemented.

In a similar vein, the hypothetical reduction performed in section 4.8 could be made concrete by rebuilding the experiment with preferential attachment as an important pillar. As postulated, this may lead to an initial scale-free network emerging and lead to the same results of Albert, Barabási and Jeong.

Bibliography

[1] M. Newman, "Power laws, pareto distributions and zipf's law," *Contemporary Physics*, vol. 46, no. 5, p. 323–351, Sep 2005. [Online]. Available: http://dx.doi.org/10.1080/00107510500052444

[2] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[3] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," 2007.

[4] A. D. Broido and A. Clauset, "Scale-free networks are rare," *Nature Communications*, vol. 10, no. 1, Mar 2019. [Online]. Available: http://dx.doi.org/10.1038/s41467-019-08746-5

[5] A.-L. Barabási and M. Pósfai, *Network science*. Cambridge: Cambridge University Press, 2016. [Online]. Available: http://barabasi.com/networksciencebook/

[6] A.-L. Barabasi and E. Bonabeau, "Scale-free networks," , vol. 4, p. 00, 01 2010.

[7] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, no. 1, p. 47–97, Jan 2002. [Online]. Available: http://dx.doi.org/10.1103/RevModPhys.74.47

[8] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, p. 509–512, Oct 1999. [Online]. Available: http://dx.doi.org/10.1126/science.286.5439.509

[9] M. Bonani, V. Longchamp, S. Magnenat, P. Rétornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada, "The marxbot, a miniature mobile robot opening new perspectives for the collective-robotic research," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 4187–4193.

[10] J. F. Roberts, T. S. Stirling, J.-C. Zufferey, and D. Floreano, "2.5d infrared range and bearing system for collective robotics," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, ser. IROS'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 3659–3664. [Online]. Available: http://dl.acm.org/citation.cfm?id=1732643.1732651

[11] C. Pinciroli, "Argos3," https://github.com/ilpincy/argos3, 2019.

[12] I. Rausch, A. Reina, P. Simoens, and Y. Khaluf, "Coherent collective behaviour emerging from decentralised balancing of social feedback and noise," *Swarm Intelligence*, vol. 13, no. 3, pp. 321–345, Dec 2019. [Online]. Available: https://doi.org/10.1007/s11721-019-00173-y

[13] A. Czirók, A.-L. Barabasi, and T. Vicsek, "Collective motion of self-propelled particles: Kinetic phase transition in one dimension," *Physical Review Letters*, vol. 82, 01 1998.

[14] J. Buhl, D. J. T. Sumpter, I. D. Couzin, J. J. Hale, E. Despland, E. R. Miller, and S. J. Simpson, "From disorder to order in marching locusts," *Science*, vol. 312, no. 5778, pp. 1402–1406, 2006. [Online]. Available: https://science.sciencemag.org/content/312/5778/1402

[15] C. Huepe, G. Zschaler, A.-L. Do, and T. Gross, "Adaptive-network models of swarm dynamics," *New Journal of Physics*, vol. 13, no. 7, p. 073022, Jul 2011. [Online]. Available: http://dx.doi.org/10.1088/1367-2630/13/7/073022

[16] J. Alstott, E. Bullmore, and D. Plenz, "powerlaw: A python package for analysis of heavy-tailed distributions," *PLoS ONE*, vol. 9, no. 1, p. e85777, Jan 2014. [Online]. Available: http://dx.doi.org/10.1371/journal.pone.0085777

[17] T. Nepusz, "plfit: Fitting power-law distributions to empirical data," https://github.com/ntamas/plfit, 2019.

[18] A. Clauset, M. Young, and K. S. Gleditsch, "On the frequency of severe terrorist events," *Journal of Conflict Resolution*, vol. 51, no. 1, p. 58–87, Feb 2007. [Online]. Available: http://dx.doi.org/10.1177/0022002706296157

[19] Q. H. Vuong, "Likelihood ratio tests for model selection and non-nested hypotheses," *Econometrica*, vol. 57, no. 2, pp. 307–33, 1989. [Online]. Available: https://EconPapers.repec.org/RePEc:ecm:emetrp:v:57:y:1989:i:2:p:307-33

[20] R. Albert, H. Jeong, and A.-L. Barabási, "Diameter of the world-wide web," *Nature*, vol. 401, no. 6749, p. 130–131, Sep 1999. [Online]. Available: http://dx.doi.org/10.1038/43601

Appendix

Appendix A - Source code

In order to replicate the experiment, it is required to install ARGoS first through Carlo Pinciroli's repository at `https://github.com/ilpincy/argos3` which includes an installation guide in its readme.

ARGoS requires a Linux operating system to be able to run. During the work done on this thesis, the Linux subsystem of Windows was used to be able to run it without having to switch to a virtual machine. A recommended guide of setting up this subsystem can be found at `http://www.terriblesysadmin.com/?p=76`.

In order to setup the experiment correctly, it is recommended to install the ARGoS examples first as described in `https://github.com/ilpincy/argos3-examples`. Clone this repository as a subdirectory of the main ARGoS repo, and test that running ARGoS still works. Finally, the source code related to the experiment and various scripts used to generate the data sets can be found at `https://github.com/timothythiecke/ArgosThesis`[1]. Either clone or download these files and overwrite them in their respective folders, rebuild the projects and run the experiment with:

```
argos3 -c experiments/marching_large.argos
```

Appendix B - Data sets

Certain parts of the thesis discuss the results of scripts on data sets. They can be found in the datasets folder of the repository or directly through `https://github.com/timothythiecke/ArgosThesis/tree/master/datasets` in order to verify the presented results. The datasets folder contains subfolders per section that discussed them.

Appendix C - Power law distribution checker script

The following bash scripts allow you to check if a distribution in a file follows a powerlaw. This requires the installation of the powerlaw [16] and plfit [17] packages.

CheckPL

```
#!/bin/bash
INPUT=$1;
```

[1] self: the repo is currently private, make public or move to public clean repo

```
OUTPUT=output.txt
PVALTEST=0.1;

#~ First, check run the p-value test
plfit -p exact $INPUT > res;

#~ Second, check whether alpha or m/n (i.e. the data ratio) are 'nan'
check=$(grep 'alpha = ' res | cut -d' ' -f3);
datRat=$(grep 'm/n = ' res | cut -d' ' -f3);
pVV2=1;
if [ "$check" == "nan" ]; then
        pVV2=0;
elif [ "$check" == "-nan" ]; then
        pVV2=0;
fi

#~ Third, save the p-value in pV and check if the p-value is >= 0.1
pV=$(grep 'p= ' res | cut -d' ' -f2);
pVV=$(echo $pV'>='$PVALTEST | bc -l);

#~ Fourth, if both conditions are satisfied, write the p-value to OUTPUT
if [ "$pVV" -gt "0" ] && [ "$pVV2" -gt "0" ]; then echo p=$pV > $OUTPUT;
        echo dataRatio=$datRat >> $OUTPUT;
        python -W ignore plfit.py $INPUT $OUTPUT;
        cat $OUTPUT;
        python check_PL_output.py $INPUT
else
        echo "--- NOT POWERLAW ---"
fi;
```

plfit

```
from sys import argv

script, filename, plResults = argv

import powerlaw

with open (filename, 'r') as f:
```

```
        content = f.readlines()
content = [x.strip() for x in content]

content = [float(x) for x in content]

data = content

results = powerlaw.Fit(data,verbose=False)

resFile = open(plResults, 'a')

alpha=results.power_law.alpha
xmin=results.power_law.xmin
resFile.write("alpha=%f \nxmin=%f \n" % (alpha, xmin))

distributions = ['exponential', 'stretched_exponential', 'truncated_power_law',
↪  'lognormal', 'lognormal_positive', 'normal']

checker = False

for dist in distributions :
        R, p = results.loglikelihood_ratio('power_law', dist,
        ↪  normalized_ratio=True)
        if p <= 0.1 :
                checker = True
                resFile.write("%r \t %f \t %f \n" % (dist, R, p))
```

checkPLoutput

```
from sys import argv

script, input_file = argv

import powerlaw

with open (input_file, 'r') as f:
        content = f.readlines()
content = [x.strip() for x in content]
```

```
content = [float(x) for x in content]

data = content

results = powerlaw.Fit(data,verbose=False)
alpha=results.power_law.alpha
xmin=results.power_law.xmin

distributions = ['exponential', 'stretched_exponential', 'truncated_power_law',
↪  'lognormal', 'lognormal_positive', 'normal']

rank = 1.0
rankZero = 1.0
rankTPL = 1.0
rankZeroTPL = 1.0

for dist in distributions :
        R, p = results.loglikelihood_ratio('power_law', dist,
        ↪  normalized_ratio=True)
        if p <= 0.1 :
                if R < 0:
                        rankZero = 0.0
                elif R > 0:
                        rank += 1.0

for dist in distributions :
        R, p = results.loglikelihood_ratio('truncated_power_law', dist,
        ↪  normalized_ratio=True)
        if p <= 0.1 :
                if R < 0:
                        rankZeroTPL = 0.0
                elif R > 0:
                        rankTPL += 1.0


rank = rankZero * rank / 7.0
rankTPL = rankZeroTPL * rankTPL / 7.0
rankTowrite=rank
Lambda=0.0
```

```
checkDist="n"
if (rank > 0.0 or rankTPL > 0.0):
        if rank > rankTPL:
                checkDist="pl"
        elif rank < rankTPL:
                checkDist="tpl"
                rankTowrite=rankTPL
                alpha=results.truncated_power_law.alpha
                Lambda=results.truncated_power_law.Lambda
        else:
                checkDist="pl-tpl"

if rankTowrite > 0.0:
        print("--- OK ---")
else:
        print("--- NOT POWERLAW ---")
```